D.8

CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS

RECOMMENDATION FOR SPACE DATA SYSTEM STANDARDS:

PACKET TELEMETRY

"BLUE BOOK"

MAY 1984

CCSDS

DEDICATION

This document is dedicated to the memory of Mr. Michel F. Pellet of the European Space Agency. His vision and leadership were an inspiration to many technical personnel who contributed to this Recommendation. He will be deeply missed by many members of the CCSDS, and by all people who value the increased international understanding which is achieved through cooperation in the peaceful scientific exploration of space.

AUTHORITY

Issue:	Blue Book, Issue-l.
Date:	May 1984.
Location:	CCSDS Plenary Meeting, May 1984, Tokyo, Japan.

This Recommendation reflects the consensus technical agreement of the following member Agencies of the Consultative Committee for Space Data Systems (CCSDS):

- Centre National D'Etudes Spatiales (CNES)/France.
- Deutsche Forschungs-u. Versuchsanstalt fuer Luft und Raumfahrt e.V (DFVLR)/West Germany.
- European Space Agency (ESA)/Europe.
- Indian Space Research Organization (ISRO)/India.
- Instituto de Pesquisas Espaciais (INPE)/Brazil.
- National Aeronautics and Space Administration (NASA)/USA.
- National Space Development Agency of Japan (NASDA)/Japan.

The following observer Agencies also concur with this Recommendation:

- Department of Communications, Communications Research Centre (DOC-CRC)/Canada.
- Institute of Space and Astronautical Science (ISAS)/Japan.
- Radio Research Laboratories (RRL)/Japan.

This Recommendation is published and maintained by:

CCSDS Secretariat
Communications and Data Systems Division
Code-TS
National Aeronautics and Space Administration
Washington
DC 20546
USA

STATEMENT OF INTENT

The Consultative Committee for Space Data Systems (CCSDS) is an organization officially established by the management of seven member space Agencies. The Committee meets periodically to address data systems problems that are common to all participants, and to formulate sound technical solutions to these problems. Inasmuch as participation in the CCSDS is completely voluntary, the results of Committee actions are termed RECOMMENDATIONS and are not considered binding on any Agency.

This RECOMMENDATION is issued by, and represents the consensus of, the CCSDS Plenary body. Agency endorsement of this RECOMMENDATION is entirely voluntary. Endorsement, however, indicates the following understandings:

0 Whenever an Agency establishes a CCSDS-related STANDARD, this STANDARD will be in accord with the relevant RECOMMENDATION. Establishing such a STANDARD does not preclude other provisions which an Agency may develop.

0 Whenever an Agency establishes a CCSDS-related STANDARD, the Agency will provide other CCSDS member Agencies with the following information:

-- The STANDARD itself.

-- The anticipated date of initial operational capability.

-- The anticipated duration of operational service.

0 Specific service arrangements shall be made via memoranda of agreement. Neither this RECOMMENDATION nor any ensuing STANDARD is a substitute for a memorandum of agreement.

No later than five years from its date of issuance, this Recommendation will be reviewed by the CCSDS to determine whether it should: (1) remain in effect without change: (2) be changed to reflect the impact of new technologies, new requirements, or new directions; or, (3) be retired or cancelled.

FOREWORD

This document is a technical Recommendation for use in developing packetized telemetry systems and has been prepared by the Consultative Committee for Space Data Systems (CCSDS). The Packet Telemetry concept described herein is the baseline concept for spacecraft-to-ground data communication within missions that are cross-supported between Agencies of the CCSDS.

This Recommendation establishes a common framework and provides a common basis for the data structures of spacecraft telemetry streams. It allows implementing organizations within each Agency to proceed coherently with the development of compatible derived Standards for the flight and ground systems that are within their cognizance. Derived Agency Standards may implement only a subset of the optional features allowed by the Recommendation and may incorporate features not addressed by the Recommendation.

Through the process of normal evolution, it is expected that expansion, deletion or modification to this document may occur. This Recommendation is therefore subject to CCSDS document management and change control procedures which are defined in Reference [1].

DOCUMENT CONTROL

Issue	Title	Date	Status/Remarks
1	Recommendation for Space Data Systems Standards: Packet Telemetry.	May 1984	Original Issue

CONTENTS

Sections

Figures

Annexes

REFERENCES

[1] "Procedures Manual for the Consultative Committee for Space Data Systems', CCSDS Document, May 1984 or later issue.

[2] 'Recommendation for Space Data System Standards: Telemetry Channel Coding", CCSDS Blue Book, May 1984 or later issue.

[3] "Recommendation for Space Data System Standards: Telecommand, Part-2: Transfer Frame Pormats and Procedures., CCSDS document, April 1984 or later issue.

[4] "Recommendation for Space Data System Standards: Time Code Formats., CCSDS document, February 1984 or later issue.

The latest issues of these documents may be obtained from the CCSDS Secretariat at the address indicated on Page-ii.

1 INTRODUCTION

1.1 PURPOSE

The purpose of this document is to establish a common Recommendation for the implementation of spacecraft "Packet Telemetry" systems by the Agencies participating in the Consultative Committee for Space Data Systems (CCSDS).

1.2 SCOPE

Packet Telemetry is a concept which facilitates the transmission of space-acquired data from source to user in a standardized and highly automated manner. Packet Telemetry provides a mechanism for implementing common data structures and protocols which may enhance the development and operation of space mission systems.

This Recommendation addresses the following two processes:

(1) The end-to-end transport of space mission data sets from source application processes located in space to distributed user application processes located on the ground.

(2) The intermediate transfer of these data sets through space data acquisition networks, which contain spacecraft, radio links, tracking stations, ground communications circuits and mission control centers as some of their components.

This Recommendation is limited to describing the telemetry formats which are generated by the spacecraft in order to execute its role in the above processes. The jointly-agreed CCSDS channel coding mechanisms required to implement space-to-ground data links of acceptable quality are defined in Reference [2]. This Recommendation therefore contains specifications for the following data structures:

(a) A SOURCE PACKET, which provides protocol data formatting services so that data may be exchanged between a source application process in space and its associated user application process(es) on the ground. The Source Packet format is defined in Section 3. Optional SEGMENTATION data structures are also described which permit very long Source Packets to be reformatted into shorter data units. The Segmentation options are defined in Section 4.

(b) A **TRANSFER FRAME**, which facilitates movement of the packetized or segmented source data through the spacecraft-to-ground communications path. As an alternative to Segmentation, the Transfer Frame also provides a mechanism to time-share the data link between sources by creating logical **VIRTUAL CHANNELS..** The Transfer Frame format is defined in Section 5.

1.3 APPLICABILITY

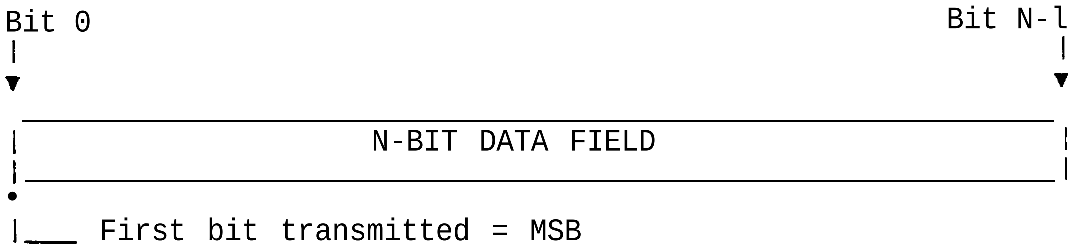
This Recommendation applies to the future exchange of Packet Tel emet ry between CCSDS Agencies in cross-support situations. The Recommendation includes comprehensive specification of the structure of data streams that are generated by remote space vehicles for telemetering to space mission data processing facilities (which are usually located on Earth). The Recommendation does not attempt to define the architecture or configuration of these data processing facilities, except to describe assumed ground data handling services which affect the selection of certain onboard formatting options.

The Recommendation specifies a wide range of formatting capabilities which may facilitate a high degree of flexibility in the design of spacecraft data acquisition systems; how ever, compatibility with the Packet Telemetry concept may be realized by only implementing a narrow subset of these capabilities. Some “Application Notes” which discuss how different levels of compatibility may be achieved are included in Annex E.

1.4 BIT NUMBERING CONVENTION AND NOMENCLATURE

The following "Caution" should be observed when interpreting the bit numbering convention which is used throughout this CCSDS Recommendation:

> CAUTION
>
> In this document, the following convention is used to identify each bit in a forward justified N-bit field.
>
> The first bit in the field to be transmitted (i.e. the most left justified when drawing a figure) is defined to be "Bit 0"; the following bit is defined to be "Bit 1" and so on up to "Bit N-l". When the field is used to express a binary value (such as a counter), the Most Significant Bit (MSB) shall be the first transmitted bit of the field, i.e. "Bit 0".



In accordance with modern data communications practice, spacecraft data fields are often grouped into 8-bit "words" which conform to the above convention. Throughout this Recommendation, the following nomenclature is used to describe this grouping:

"8-bit word" = "Octet"

2 OVERVIEW

Figure 2-1 is a functional diagram of the telemetry data flow from the creation of a data set by an application process operating within a spacecraft "source" (instrument or subsystem), through to the delivery of the same data set to a user "sink" (application process) on the ground. Since many of the elements of this flow are presently mission unique, a primary objective of Packet Telemetry is to define stable, mission- independent interface standards for the communications paths within the flow.

Data structures within a packet telemetry system are provided:

(a) To allow the user to optimize the size and structure of his application data set with a minimum of constraints imposed by the spacecraft-to-ground transport system. The user should thus be able to define his data organization independently of other users and to adapt this organization to the various modes of his experiment .

The data structure which enables this independence is the telemetry Source Packet. User data are encapsulated within a packet by prefacing them with a standard label or "primary header", which is used by the data transport system to route the data through the system. A secondary header structure is provided which enables more application-unique specification of the user data.

(b) To allow the ground data acquisition system to capture the packets in a standardized way, with a data-independent method of performing packet reassembly and determination of data quality. The data structure within the packet telemetry system for achieving this is the Transfer Frame, which provides the necessary header elements for extracting packets or segments from the frame and allows optional error-detection coding for deciding whether the frame handling process was error free.

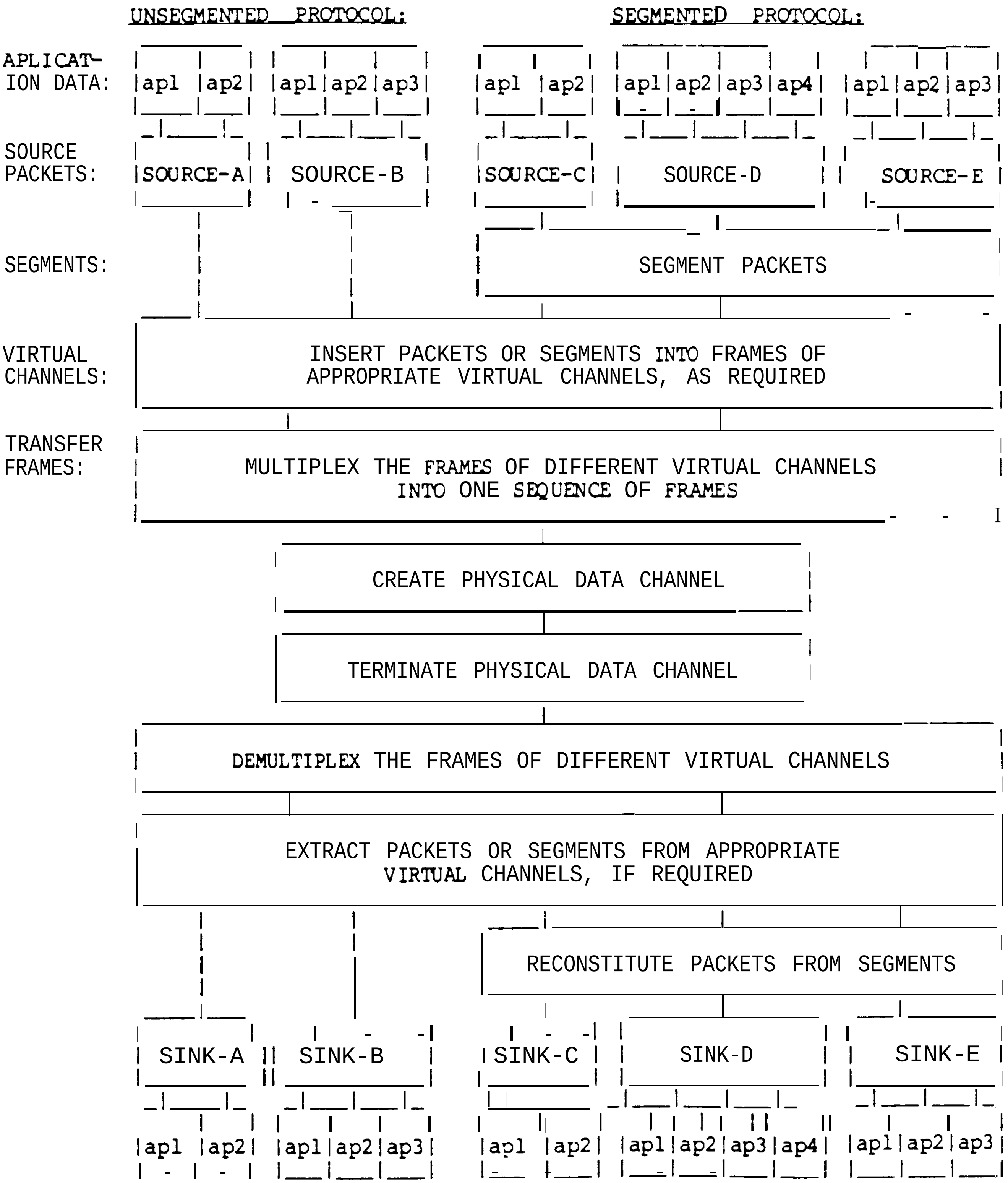

UNSEGMENTED PROTOCOL:
SEGMENTED PROTOCOL:
APLICAT-ION DATA:
ap1
ap2
ap1
ap2
ap3
ap1
ap2
ap1
ap2
ap3
ap4
ap1
ap2
ap3
SOURCE PACKETS:
SOURCE-A
SOURCE-B
SOURCE-C
SOURCE-D
SOURCE-E
SEGMENTS:
SEGMENT PACKETS
VIRTUAL CHANNELS:
INSERT PACKETS OR SEGMENTS INTO FRAMES OF APPROPRIATE VIRTUAL CHANNELS, AS REQUIRED
TRANSFER FRAMES:
MULTIPLEX THE FRAMES OF DIFFERENT VIRTUAL CHANNELS INTO ONE SEQUENCE OF FRAMES
CREATE PHYSICAL DATA CHANNEL
TERMINATE PHYSICAL DATA CHANNEL
DEMULTIPLEX THE FRAMES OF DIFFERENT VIRTUAL CHANNELS
EXTRACT PACKETS OR SEGMENTS FROM APPROPRIATE VIRTUAL CHANNELS, IF REQUIRED
RECONSTITUTE PACKETS FROM SEGMENTS
SINK-A
SINK-B
SINK-C
SINK-D
SINK-E
ap1
ap2
ap1
ap2
ap3
ap1
ap2
ap1
ap2
ap3
ap4
ap1
ap2
ap3

FIGURE 2-1: TELEMETRY DATA FLOW

(c) To allow the spacecraft terminus of the data transport system to be designed and tested without a detailed definition of all user data organizations. This is particularly important since at the time of specifying the onboard spacecraft data system design this knowledge is usually not available from the experimenter, and attempts to force an early agreement with present systems of ten lead to non-optimum fixed-format telemetry design.

Since most space communications systems are capacity-limited, multiple users must be guaranteed access to the downlink data channel. It is therefore important for the spacecraft to be able to manage the data flow to the ground in an orderly manner. For example, users who generate long packets could, if allowed unrestricted access, monopolize the channel for long periods of time, forcing increased buffer capacity and response time for other users. This problem is solved by permitting two methods for controlling data flow, namely:

(i) Virtual Channelization: This is a logical mechanism that allows sources which generate very long packets to be "virtually" given exclusive access to the physical data channel by assigning them transmission capacity on a frame-by-frame basis. The data structure used to implement this feature is the "Virtual Channel..

Virtual channelization will normally be used to separate sources of very different characteristics. For example, if a payload contains an imaging instrument which produces a regular scan line packet containing many thousands of bits, and a number of processor-based experiments which aperiodically generate smaller packets of processed data, a possible system architecture would be to assign the imaging instrument to one virtual channel and to handle the rest by segmentation or direct multiplexing on a second virtual channel.

(ii) Segmentation: Several optional mechanisms exist for transmitting long source packets as a series of shorter packets or segments, thus avoiding exclusive capture of the channel by one source. Within one option a formal data structure is introduced to implement this feature: this is the "Telemetry Segment..

"Application Notes", which describe how compatibility with these various data structures may be achieved, are presented in Annex E. Key elements of the rationale behind Packet Telemetry are presented in Annex F.

3 SOURCE

A Source Packet encapsulates a block of observational and ancillary application data which is to be transmitted from a data source in space to a source analysis facility on the ground.

The Source Packet structure permits future "versions" of the data structure to be defined, if required. The Version 1 Source Packet format, which is shown in Figure 3-1, consists of the following four major fields:

Major Field	Length (bits)
Primary Header	48
Secondary Header	Variable (optional)
Source Data	Variable
Packet Error Control	16 (optional)

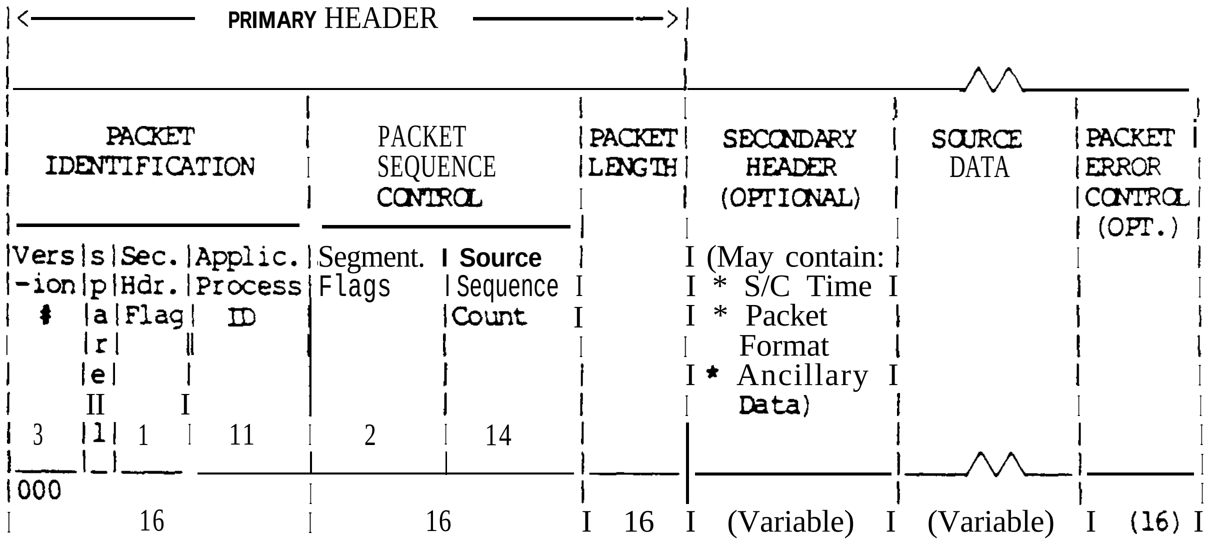


FIGURE 3-1: VERSION 1 'SOURCE PACKET. FORMAT

3.1 PRIMARY HEADER

The primary header consists of 48 bits subdivided into the following fields:

Field	Length (bits)
PACKET IDENTIFICATION	16
- Version Number (3)	
- Reserved Spare (1)	
- Secondary Header Flag (1)	
- Application Process ID (11)	
PACKET SEQUENCE CONTROL	16
- Segmentation Flags (2)	
- Source Sequence Count (14)	
PACKET LENGTH	16
	--
	48

3.1.1 Packet Identification (16 bits). This 16-bit field is separated into four subfields:

(a) Version Number (Bits 0 through 2).

The Version Number occupies the three most significant bits of the packet Primary Header. By changing the Version Number, future variations of the Source Packet structure become possible. AT PRESENT, ONLY TWO VERSIONS OF TEE PACKET ARE PERMITTED:

- "Version 1" (Bits O-2 = 000) is the complete telemetry Source Packet format, which is described in the remainder of Section 3.
- "Version 2" (Bits O-2 = 100) is the Telemetry Segment format, which is described in Section 4.

FOR VERSION 1, THE REMAINDER OF THE SOURCE PACKET FORMAT IS DEFINED AS FOLLOWS:

(b) Reserved Spare (Bit 3).

This single bit is reserved for future application. At present, it shall be set to "0".

(c) Secondary Header Flag (Bit 4).

This l-bit flag signals the presence (Bit 4 = 1) or absence (Bit 4 = 0) of a Secondary Header data structure within the Source Packet.

(d) Application Process ID (Bits 5 through 15).

This 11-bit field uniquely identifies the individual application process within a particular space vehicle which created the Source Packet. (Note: the space vehicle itself is identified by the Spacecraft Identifier in the Transfer Frame header.) The Application Process ID's are tailored to local mission needs and are therefore assigned by the Mission Manager. Guidelines for assigning Application Process ID's may be developed by the CCSDS. Users should note that ground data accounting considerations may limit the number of different application processes which may be simultaneously "open" during a given session.

The "all ones" configuration of the Application Process ID shall be reserved to identify "Idle Packets", which are generated by the spacecraft data system to maintain synchronism of the ground packet extraction process during periods when no sources have packetized data available for transfer to the ground.

3.1.2 Packet Sequence Control (16 bits). This 16-bit field is separated into two fields:

(a) Segmentation Flags (Bits 0,1).

The Segmentation Flags, which occupy the two most-significant bits of the 16-bit field, are used to indicate the status of long message-oriented Source Packets that have been broken into shorter communications-oriented segments. Various optional processes for segmentation are described in Section 4. The assignment of the flags is as follows:

(i) Last Segment Flag (Bit 0).

When Bit 0 is set to a "1", this indicates that the remainder of the data structure contains the last segment of a long segmented Source Packet.

(ii) First Segment Flag (Bit 1).

When Bit 1 is set to a "1", this indicates that the remainder of the data structure contains the first segment of a long segmented Source Packet.

The Segmentation Flags are thus interpreted as follows:

Bit-O:	Bit-l :	Interpretation:
0	0	Continuation Segment
0	1	First Segment
1	0	Last Segment
	1	Unsegmented Packet

Note: When a source does not perform segmentation, it shall set the flags to “11” before passing the packet to the spacecraft data handling system for transfer to the ground.

(b) Source Sequence Count (Bits 2 through 15).

This 14-bit field contains a straight sequential count (modulo 16384) of each packet generated by each unique source application process on the spacecraft. The purpose of the field is to associate this packet with other packets from the same application process, even though their natural order may have been disturbed during transport to the user’s processor on the ground. The field will normally be used in conjunction with the spacecraft measurement time code to provide completely unambiguous association; it is therefore essential that the period of the Source Sequence Count should be sufficiently long that the time code increments at least once between successive recycling of the sequence count. For continuous operation of a source application process, it is not permissible to “short cycle” the sequence count before the full counter accumulation has been reached; however, if a source’s operation is interrupted (e.g. by loss of power), the source may start a new sequence count.

3.1.3 Packet Length (16 bits). This field contains a sequential 16-bit binary count "C" of the length (in octets) of the remainder of the data structure which is enclosed between the first bit of the Secondary Header and the last bit of the packet (i.e. the last bit of the Source Data field, or the last bit of the Error Control field if this option is selected). The field is expressed as follows:

$$C = \{ \text{(Number of octets)} - 1 \}$$

Users should recognize that although very long packets are permissable, these present special problems in terms of data link monopolization, source data buffering, and network accountability during transfer across the unique channel from the spacecraft to the ground. As described in Section 4, large packets (i.e. packets which are very much longer than the data space within the Transfer Frame) may either be segmented by the transfer network or may be assigned to individual Virtual Channels”. It is anticipated that for a large number of applications, packet lengths in the range of approximately 1 to 8 kilobits will be chosen by instrument designers for flight operations. Very short packets (less than 1024-bits) become inefficient and will normally only be used by very low-rate data sources, for engineering tests and checkout, and for fill.

For flight operations, computational speed considerations within the ground data processing elements of the transfer network may dictate that there is a maximum incoming packet rate which can be handled. This may mean that the network implementation organizations within each Agency will specify a minimum size for small packets (e.g "Idle Packets") which are transmitted contiguously.

From the viewpoint of ground data processing efficiency, it is strongly recommended that the overall packet length should be an even number of octets.

3.2 SECONDARY HEADER

The purpose of the Secondary Header is to provide a standard means for encoding within a Source Packet any ancillary data (time, internal data field format, spacecraft position/attitude, etc.) which may be necessary for the interpretation of the information contained within the packet. The presence or absence of a Secondary Header is signalled by the Secondary Header Flag within the Packet Identification field, as described in Section 3.1.1 (c). If present, the Secondary Header will contain data generated by one or more onboard application processes.

Short-term packet management processes within the ground network, such as extracting and delivering individual packets to a user in near-real time, may be able to use only the sequence count field within the packet header for accounting purposes. However, for longer-term services to be provided (e.g archiving, sorting, processing and correlation with other data sets) the sequence count must be concatenated with a "time" field in order to unambiguously identify a packet. For users needing these long-term services, it is a requirement that the Secondary Header must be present in every packet produced by the source and must contain a time code which is registered with respect to some known event encoded within the Source Data section of the packet, accurate to the time resolution required for the Source Data interpretation and association.

The length of the Secondary Header shall always occur in integral multiples of octets. Guidelines for formatting the Secondary Header are presented in Annex C.

3.3 SOURCE DATA.

The Source Data field contains the measurement information generated by the primary application process operating within each source. The only formal restriction imposed on the Source Data field is that the total length of this section must be an integral number of octets: otherwise, the experimenter will normally have complete freedom to specify the data content and the internal format of this field. How ever, users are cautioned that if the packet contents are to be processed within CCSDS Agency support facilities, then local standards for internal formatting may be imposed.

3.4 PACKET ERROR CONTROL (Optional)

At the discretion of the user, an optional error detection code may be appended to the packet in order to verify that the overall integrity of the message has been preserved during the transport process. In those configurations where the supporting Agency agrees to check the Packet Error Control code associated with a particular application process, the field shall be present in every packet produced by that application process, shall be 16-bits long and the recommended encoding polynomial is specified in Annex D; otherwise, the selection of the encoding polynomial, and the length of the field, is left to the user or to local Agency standards. The presence or absence of packet error control will be implied by the Application Process ID.

4 PACKET SEGMENTATION

The recommendations for Source Packet formatting permit a wide range of packet lengths to be implemented. Furthermore, sources may conceptually vary the packet length on a dynamic basis, according to the message formatting needs of different application processes operating within the source.

Space communications systems are usually heavily driven by the bandwidth or capacity constraints of the unique data channel which connects the spacecraft to the ground. Since multiple users must share this communications channel, flow control becomes critical to ensure that all sources have access to this common resource for periods of time consistent with their delivery timeliness requirements and their capacity to buffer data while other sources are being serviced. Very long Source Packets therefore present a flow control problem since they may monopolize the channel for unacceptable periods of time and may force other sources to implement unreasonably large local buffering.

Several mechanisms are provided for solving this flow control problem within Packet Telemetry systems. One involves assigning long-packet generating sources to their own "Virtual Channel" by inserting them into dedicated types of Transfer Frames. These dedicated frames may then be interleaved with other frames containing multi-user data which are formatted into mutually compatible packet sizes. This Virtual Channel solution is discussed in Section 5.

Another mechanism involves the use of various "segmentation" protocols whereby the spacecraft data handling system (or the source itself) breaks the long packets into shorter pieces which are compatible with the flow control requirements of other users. The long-packet source will therefore either be responsible for performing its own communications-oriented segmentation or for buffering its own data while the spacecraft breaks the packet up into smaller segments and interleaves them with similar-sized segments of data from other sources for transfer to the ground.

This issue of the Packet Telemetry Recommendation permits the system designer to select between three valid methods of segmentation:

1. Segmentation by the source application process, using the standard Version 1 "Source Packet" data structure (Section 4.1).

2. Segmentation by the spacecraft data system, using the standard Version 1 "Source Packet" data structure (Section 4.2).

3. Segmentation by the spacecraft data system, using the standard Version 2 "Telemetry Segment" data structure (Section 4.3).

These three methods are discussed below. Virtual Channelization is discussed in Section 5. The attributes of all four methods are summarized in Annex-G. Selection of the specific option(s) which will be cross-supported by CCSDS Agencies will be defined within the detailed cross-support agreements.

4.1 SOURCE-INTERNAL SEGMENTATION.

A first option is for the source application process to be constrained by the system designer to always generate packets which are short enough to ensure that no flow control problems can exist when they are integrated with packets from other onboard sources; however, this may artificially constrain the measurement acquisition processes by locking them to the communication processes. For instance, an imaging scan line may be a “natural” packet for an instrument, but it may be too long to transmit unsegmented.

A second option is to allow the source application process to format measurement data into very long packets in accordance with sampling needs, but to require that the source interface electronics breaks them into shorter segments prior to delivery to the spacecraft data system. This may be accomplished by having the source reformat the long Version 1 packets into shorter Version 1 packets within which the Segmentation Flags (Section 3.1.2a) are manipulated to signal a first, continuing, or last segment of the larger packet entity. Within this option, the “Source Sequence Count” field must increment once for each segment generated and the “Packet Length” field must directly indicate the length of the segment; therefore, information which describes the Sequence Count and Length of the original large packet should be encoded within the data field of at least the first segment.

4.2 SPACECRAFT SEGMENTATION USING SOURCE PACKETS.

A third option exists whereby the spacecraft data system implements one or more special internal application processes which are dedicated to the task of segmenting long packets from onboard sources. Within this option, the entire user packet is recursively treated as “data” in a new Version 1 packet, which bears Packet Identification, Packet Sequence Control and Packet Length fields that are unique to the spacecraft data system application process which performs the segmentation. In this case, the Segmentation Flags (Section 3.1.2 a) of the new packet shall be manipulated to signal a first, continuing or last segment of the original user packet. This solution introduces a modest amount of extra overhead, since the first segment in such a system will contain the Primary Header of the spacecraft data system segmentation process, followed by the complete Primary Header of the packet which is being segmented.

4.3 SPACECRAFT SEGMENTATION USING "TELEMETRY SEGMENTS".

All three of the above options for.segmentation utilize the Version 1 Source Packet format, within which the “length” field is always interpreted as the length in octets of the data unit (packet or segment) that appears on the downlink and the "sequence count” increments once per data unit that is transmitted from a given application process. A fourth option exists, using “Version 2” of the packet format (see Section 3.1.1a), within which the “length” field in the data unit defines the length of an original packet that remains to be transmitted and the “sequence count” is static since it refers to the numbering of the original packet by the application process. The length and sequence count of the data unit being transmitted are therefore semantically different between the two versions, since these parameters are implied when using Version 2.

The Version 2 “Telemetry Segment” structure is shown in Figure 4-l:

SEGMENT HEADER							
SEGMENT IDENTIFICATION				SEGMENT SEQUENCE CONTROL		SEGMENT LENGTH	
Vers-ion #	Spare	Sec. Hdr. Flag	Applic. Process ID	Segment. Flags	Original Packet Sequence Count	(Length of the original Source Packet which remains yet to be transmitted)	SEGMENTED SOURCE PACKET DATA
3 100	1	1	11	2	14		
16				16		16	2048, 4096 or 8192

FIGURE 4-1: VERSION 2 "TELEMETRY SEGMENT" FORMAT

The semantic definition of the Version 2 "Segment Header" differs from the Version 1 (Source Packet) "Primary Header" *in* two important ways:

1. The Packet Sequence Count field is static within all the segments associated with a long packet which is being segmented since it contains the count of the original packet.

2. The Segment Length field does not indicate the length of the segment, but instead indicates the length of data from the original long packet (including that contained within the segment) which remains yet to be transmitted. The length of the segment is fixed and is specified externally.

   The length of the segment of data from the original Source Packet which is being transmitted ("LSEGMENT", in octets) shall be fixed for a particular Virtual Channel on a given mission. The available fixed lengths are LSEGMENT = 256, 512 or 1024 octets (2048, 4096 or 8192 bits), with 512 octets being the preferred value since shorter lengths result in excessive overhead and ground computational speed requirements. The selected length shall be indicated in the header of the Transfer Frame, as described in Section 5.2.4(d).

   Note that the total length of each Telemetry Segment will be (LSEGMENT+6) octets since the standard 48-bit Version 2 Segment Header always precedes the segment of data.

4.3.1 Segmentation Process, When using the Version 2 Telemetry Segment to perform segmentation, the process shall be as follows:

(a) An unsegmented Version 1 Source Packet is input to the processor which performs the segmentation.

(b) In the first segment, the Version Number is modified to indicate Version 2 (i.e. "100"), and the Segmentation Flags (Bits 0 and 1 of the Packet Sequence Control field) are modified from the "11" state (unsegmented) to the "01" state (first segment), as described in Section 3.1.2(a). In the first segment, the Packet Length field is unchanged and therefore indicates the length of the original packet WHICH HAS YET TO BE TRANSMITTED, including this segment. This field is then followed by the first (LSEGMENT) octets of the Secondary Header and Data Field of the original Version 1 packet.

(c) In continuation segments, the Version Number continues to indicate Version 2, the Segmentation Flags are set to the "00" configuration and the original Packet Length field is progressively decremented by (n)x(LSEGMENT) octets as each segment is transmitted, where (n)= 1,2,3, etc. according to the sequential number of the continuation segment.

(d) In the last segment, which occurs when the original Packet Length field decremented by (n)x(LSEGMENT) octets contains a value which is less than or equal to (LSEGMENT), the Version Number remains as Version 2, the Segmentation Flags are set to "10", and the Packet Length field then directly indicates the length of the residue of the original packet which is contained within the segment.

This process is illustrated in Figure 4-2, which shows how a Version 1 Source Packet whose overall length is 2106-octets (2100-octets of Secondary Header and Source Data plus 6-octets of Primary Header) is progressively broken into four 512-octet Version 2 Telemetry Segments of overall length 518-octets (512-octets of data plus 6-octets of Segment Header) and a last segment of overall length 58-octets (52-octets of data plus 6-octets of Segment Header). Note that the convention of:

"Indicated length = { (Number of octets) - 1 }"

is used for the Segment Length field, as defined in Section 3.1.3.

It should also be noted that it is permissableto generateVersion Telemetry Segments which have the Segmentation Flags set to "11" (unsegmented): in this case the Segment Length field indicates the length of the remainder of the Segment.

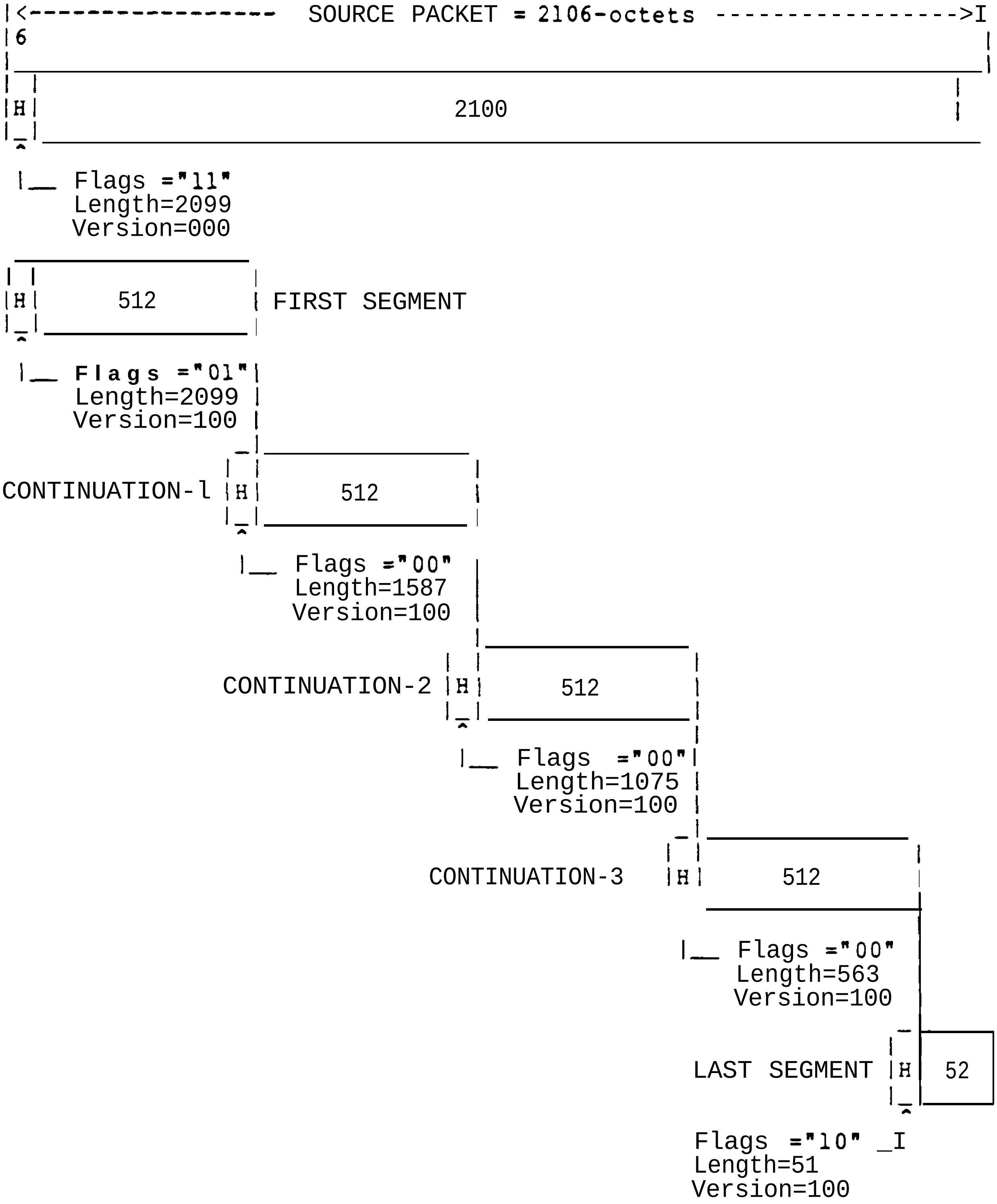


FIGURE 4-2: EXAMPLE OF THE SEGMENTATION PROCESS

Attention is drawn to the following features of this protocol:

1. It is possible to mix segmented and unsegmented Source Packets together on the same Virtual Channel, since every packet header contains a Version Number and self-specifying Segmentation Flags.

2. Within the ground processor which extracts Version 2 segments from the frame, the location of the Segment Headers may be determined from a simple test of the Last Segment flag in the Segment Sequence Control field:

   (i) If the Last Segment flag is "0", then the next header will be found by counting forward by (LSEGMENT) octets after the end of the Segment Length field.

   (ii) If the Last Segment flag is "1", then the next header will be found by counting forward by the number of octets indicated in the Segment Length field.

3. Since the fixed segment lengths are defined to be pure binary quantities (256, 512 or 1024 octets), then by implementing this decrementing length approach the most significant bits of the Segment Length field will decrease in a binary countdown fashion as successive segments are transmitted. This information provides a "serial number" for the segment which may be used to recombine segments should their natural order be disturbed during transmission. For example, with 512-octet segments, the most significant octet of the length field will form this binary down-counter. Using the same case which is shown in Figure 4-2, the Length field thus would appear as follows:

Length Bit:	00	01	02	03	04	05	06	07	|	08	09	10	11	12	13	14	15
1st. Segment:	0	0	0	0	1	0	0	0	1	0	0	1	1	0	0	1	1
Continuation-l:	0	0	0	0	0	1	1	0	|	0	0	1	1	0	0	1	1
Continuation-2:	0	0	0	0	0	1	0	0	|	0	0	1	1	0	0	1	1
Continuation-3:	0	0	0	0	0	0	1	0	|	0	0	1	1	0	0	1	1
Last Segment:	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	1	1

Counter
(= Segment
serial #)

5 TRANSFER

The Version 1 Source Packet or Version 2 Telemetry Segment data formats described in Sections 3 and 4 must be embedded within a data transfer structure for transmission across the downlink data channel which connects the spacecraft to a data capture element on the ground: this data structure is the "Transfer Frame". Multiple CCSDS standard versions of the Transfer Frame may be defined in the future; however, this issue of the Recommendation for Packet Telemetry only recognizes Version 1* of the Transfer Frame format.

The Transfer Frame draws upon a layer of noisy channel services (e.g. carrier, modulation/detection, and coding/decoding) in order to establish the downlink data path. The error probabilities attainable within the noisy channel layer depend upon many factors, including received signal-to-noise ratio and coding scheme used. Digital encoding may be applied to the data channel in order to improve the system error performance. Al though Packet Telemetry systems may be designed to tolerate channel noise in the same way that conventional systems have been designed in the past (i.e. by placing data within the frame in a predetermined sequence), full benefit from Packet Telemetry will require that a high quality data channel is provided so that packetized data may be ADAPTIVELY inserted into the frame. Reference [2] describes the CCSDS Recommendation for Telemetry Channel Coding, including specification of a convolutionally encoded inner channel concatenated with a Reed-Solomon block-oriented outer code. Although not mandatory, the recommended approach is to use the concatenated Reed-Solomon/convolutional encoding option since the data channel through which the frame is transmitted will then display virtually perfect data quality.

A mechanism must be provided to detect the presence of errors which may have been introduced within the frame during transmission through the downl ink channel. If the recommended concatenated Reed-Solomon/convolutional coding is used, the Reed-Solomon error syndrome will indicate whether or not the frame is likely to contain an error. If Reed-Solomon encoding is NOT used, each Version 1 Transfer Frame must contain an error-detecting polynomial appended within its trailing octets: this polynomial may be used to determine if a frame is likely to contain an error.

The operational procedures for handling frames which contain a detected error are beyond the scope of this Recommendation and will therefore be negotiated between the Agencies when detailed cross-support agreements are formalized. This Recommendation also recognizes that Projects may desire to implement special "emergency" telemetry modes for the transmission of critical information in the event of potentially catastrophic behaviour, such as the loss of spacecraft attitude references. If these emergency modes are beyond the scope of this Recommendation, they will also be the subject of detailed cross-support agreements.

Within a single physical data channel for a particular mission, the total length of the Version 1 Transfer Frame shall be a fixed, integral number of octets: the implemented length shall be specified to the ground network as a mission set-up parameter. Although each Agency may set specifications for the minimum length of the Version 1 Transfer Frame to be used on missions under its control, it is desirable to select a common maximum value for cross-support situations in order to avoid unnecessary variability and complexity. The maximum length has been computed for the case where the Transfer Frame is synchronously inserted into a Reed-Solomon codeblock structure. For cross-support situations, a Reed-Solomon interleave depth of five is recommended. Because the data space within this recommended (I= 5) Reed-Solomon codeblock has a fixed maximum length of 8920-bits, a standard maximum Version 1 Transfer Frame length of 8920-bits has been selected for missions which are cross-supported between CCSDS Agencies since this is compatible with synchronous insertion of the Frame into the codeblock data space. Figure 5-l illustrates how a maximum length Version 1 Transfer Frame may be synchronously embedded in the standard Reed-Solomon codeblock.

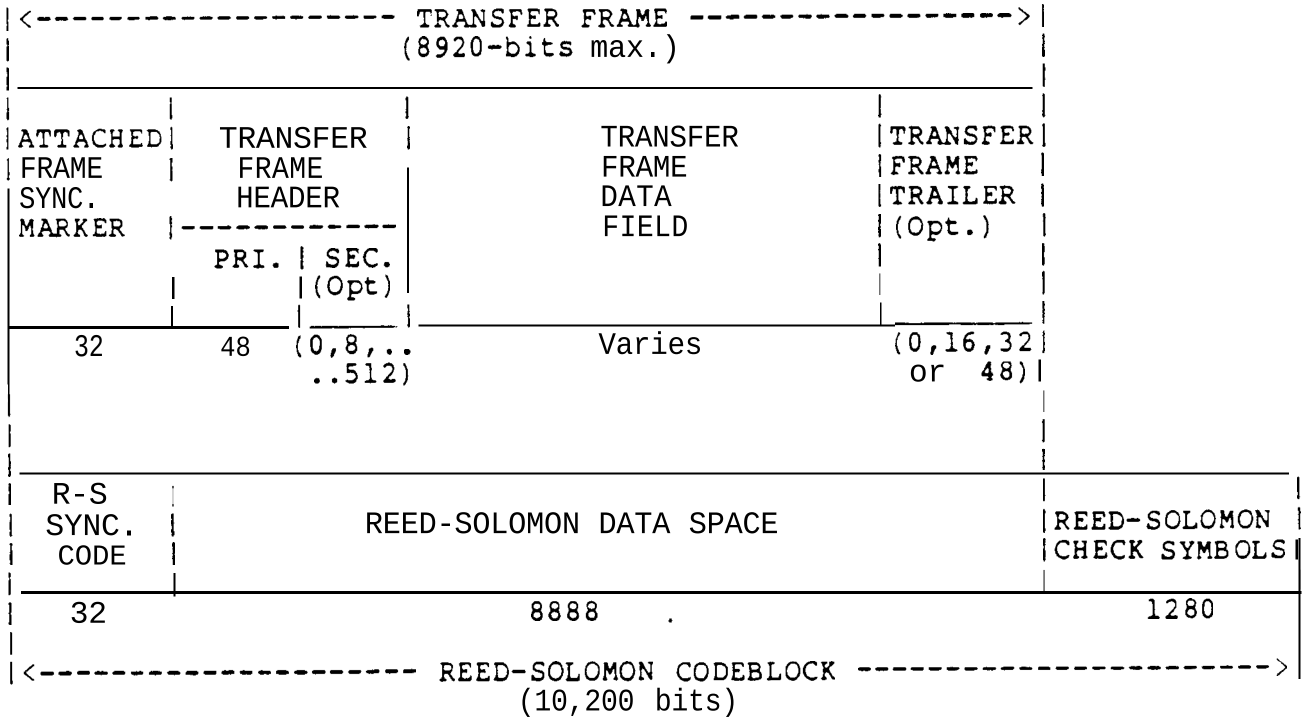


FIGURE 5-l: COMPOSITE VERSION 1 TRANSFER FRAME/ REED-SOLOMON CODEBLOCK FORMAT

The major fields of the composite Version 1 data structure shown in Figure 5-1 are as follows:

Major Field	Length (bits)	
ATTACHED FRAME SYNCHRONIZATION MARKER	32	(Note-l)
TRANSFER FRAME PRIMARY HEADER	48	
TRANSFER FRAME SECONDARY HEADER (Optional)..	(0,8,16..512)	
TRANSFER FRAME DATA FIELD	(Varies)	(Note-21
TRANSFER FRAME TRAILER (Optional)	(0,16,32,or48)	

Total Length:	Up to 8920	(Note-21

Notes:

1. As a transitional measure for the cross-support of ongoing missions, a 16-bit marker may be implemented if short frames are transmitted without Reed-Solomon coding, through a channel which displays good received signal-to-noise ratio characteristics.

2. The maximum frame length for cross-support is 8920-bits, which consists of: the Attached Synchronization Marker (32-bits); the mandatory Primary Header (48-bits); the optional Secondary Header (up to 512-bits); the Frame Data Field; and the optional Frame Trailer (16, 32 or 48-bits). The maximum length of the Data Field is therefore computed by subtracting the selected mandatory and optional components from 8920-bits. Caution: Projects which implement frame lengths shorter than the maximum, and which insert these frames synchronously into the chosen (I = 5) Reed-Solomon data space, must ensure that the correct amount of "Virtual Fill" is inferred by the ground decoder, as described in Reference [2].

5.1 ATTACHED FRAME SYNCHRONIZATION MARKER.

The Attached Frame Synchronization Marker delimits the boundaries of a fixed-length Transfer Frame. If the frame is not Reed-Solomon encoded, it is used by the ground network to acquire synchronization with the frame boundaries after transmission through the data channel. Rules for the two recognized "Cases" of attaching the Synchronization Marker to the Transfer Frame are described below:

CASE-A: When the frames are synchronously embedded within a Reed-Solomon codeblock, i.e. the frame fits exactly within the codeblock data space, then a 32-bit marker (WHICH IS COINCIDENT WITH AND IDENTICAL TO THE 32-bit REED-SOLOMON SYNCHRONIZATION CODE) shall be attached to the beginning of the frame. After Reed-Solomon decoding, the marker shall remain attached to the beginning of the frame.

CASE-B: When the frames are NOT Reed-Solomon encoded, a 32-bit marker shall be attached to the beginning of each frame. After frame synchronization has been performed by the ground network, the marker shall remain attached to the beginning of the frame.

All cross-supported missions using a packet telemetry format shall use the same frame synchronization marker to avoid the need to setup the ground-based PCM frame synchronizers in advance of every session. Care should be taken that this synchronization marker (or its reversed and/or complemented pattern) does not routinely appear in any other portion of the Transfer Frame. This does not preclude the occasional random presence of this pattern elsewhere in the frame. The marker should be chosen so that its bit pattern is different if read in reverse.

The exact pattern to be used for the Synchronization Marker in Case-A will be specified in Reference [2]. The pattern to be used for the Synchronization Marker in Case-B is specified in Annex B.

(Note: it is permissible to use the 8-bit "Master Frame Count" field in the Transfer Frame Header as a dynamic extension of the synchronization marker, for the purpose of facilitating the frame synchronization process).

5.2 TRANSFER FRAME PRIMARY HEADER.

The Transfer Frame Header is split into "Primary" and "Secondary" elements. The mandatory Primary Header provides services which are common to all missions. The optional Secondary Header, which is described in Section 5.3, permits the header information to be tailored to match the needs of different mission classes. The composite Transfer Frame structure, containing the Primary Header and Secondary Headers, plus the Frame Data Field and optional Frame Trailer, is shown in Figure 5-2.

```
      |<----------------  TRANSFER FRAME PRIMARY HEADER  ------------------>|
      |                                                                      |
______|______________________________________________________________________|
|SYNC|       FRAME       |MASTER  |VIRTUAL|              FRAME               |
|MARK| IDENTIFICATION    |CHANNEL |CHANNEL|           DATA FIELD             |
|    |                   |FRAME   |FRAME  |             STATUS               |
|    |-------------------|COUNT   |COUNT  |----------------------------------|
|    |Ver|S/C|Virt|Oper.|        |       |Sec.   |Sync|Packet|Segment|First  |
|    | # |ID |Chan|Ctrl.|        |       |Header |Flag|Order |Length |Header |
|    |   |   |ID  |Field|        |       |Flag   |    |Flag  |ID     |Pointer|
|    |   |   |    |Flag |        |       |       |    |      |       |       |
|    |   |   |    |     |        |       |       |    |      |       |       |
|    | 2 |10 | 3  | 1   |        |       |  1    | 1  |  1   |   2   |  11   |
|____|___|___|____|_____|________|_______|       |    |      |_______|_______|
 32  |       16         |   8    |   8   | - - - - - - 16                   /
                                                                           /
                                                                          /
   ....................................................................../
   /
  /
 /
/
|
|     TRANSFER FRAME                    TRANSFER               TRANSFER
|<--  SECONDARY HEADER  -->|<------      FRAME      ------>|FRAME TRAILER |
|       (OPTIONAL)         |           DATA FIELD          |  (OPTIONAL)  |
|                          |                               |              |
|__________________________|______________/\/\_____________|______________|
| (SECONDARY  | (SECONDARY |                               |(Oper. |(Frame|
|HEADER ID)   |HEADER WTA) |                               | Ctrl. | Error|
|             |            |                               | Field)| Ctrl.|
|-------------|            |          (Spacecraft          |       | Word)|
|Sec   |Sec   |            |           Application         |       |      |
|Header|Header|            |           Data)               |       |      |
|Vers. |Length|            |                               |       |      |
|Number|      |            |                               |       |      |
|      |      |            |                               |       |      |
| (2)  | (6)  |(Up to 504) |             (Var)             | (32)  | (16) |
|      |      |____________|______________/\/\_____________|_______|______|
 - - - - - -
   ||
   ||
   \/                ____________
  "00"   ==>        | REAL-TIME  |
                    |DATA INSERT |   ==> (See Section 5.3)
                    |____________|
```

FIGURE 5-2: VERSION 1 TRANSFER FRAME HEADER FORMAT

The Primary Header performs four principal functions:

(a) Identifying the Version of the Transfer Frame which is in use.

(b) Identifying the spacecraft which transmitted the telemetered data.

(c) Switching of the single physical data channel so that it may be logically multiplexed into several "Virtual Channels", and providing accounting mechanisms to detect missing frames.

(d) Providing pointers and other control information so that variable-length Source Packet or Segment data may be extracted from the fixed-length Frame Data Field.

The VERSION 1 Transfer Frame Primary Header is composed of the following fields:

Major Field	Length (bits)
Frame Identification	16
- Version Number (2)	
- Spacecraft ID (10)	
- Virtual Channel ID (3)	
- Operational Control Field Flag (1)	
Master Channel Frame Count	8
Virtual Channel Frame Count	8
Frame Data Field Status	16
- Secondary Header Flag (1)	
- Synchronization flag (1)	
- Packet Order flag (1)	
- Segment Length ID (2)	
- First Header Pointer (11)	
	48

5.2.1 Frame Identification (16 bits). The purpose of this field is to identify which operational spacecraft created the frame of data, and to indicate if the physical data link is logically switched to form "Virtual Channels". The field is broken into four subfields:

(a) Version Number (Bits 0,1).

These two bits (which occupy the two most significant bits of the sixteen bit field) are reserved for potential evolution of the Transfer Frame structure. One possible future use of this capability could be to extend the Spacecraft ID word if all ten bits become assigned. At present, only Version1 of the Frame Header is recognized (Bits **0,1 = "00")**. The format of the remainder of the Version 1 header is as follows:

(b) Spacecraft Identifier (Bits 2 through 11).

These ten bits provide positive identification of the spacecraft node which created the frame of data. Different spacecraft identifiers will be assigned for flight vehicles, for developmental vehicles which are using the ground networks during prelaunch test operations, and for simulated data streams. The Secretariat of the Consultative Committee for Space Data Systems assigns spacecraft identifiers, as described in Reference [1].

(c) Virtual Channel Identifier (Bits 12,13,14).

The concept of using a "Virtual Channel" as a solution to the problem of long Source Packet segmentation was outlined in the introduction to Section 4. The concept may be implemented by defining different logical "types" of Transfer Frames, each of which is separately identified as a different Virtual Channel. Long Source Packets may thus be inserted only into their own dedicated "L" frame type, while shorter multi-user packets may be multiplexed together into a different "M" frame type. The composite physical stream of downlink Transfer Frames may then be created by interleaving logical frames with different "L" and "M" Virtual Channel identifiers. The effect on the long packet generating source will be similar to the Segmentation process described in Section 4, i.e. the packet will be broken into fixed blocks, the length of which will equal the length of the Data Field within the Transfer Frame. During times when the other source data are being transmitted within their "M" Type of frame, the long packet source data will be buffered until an opportunity occurs for the spacecraft to interleave an "L" Type frame containing a segment of that packet.

The Virtual Channel facility also allows complete frames of data from other spacecraft generating sources (e.g. tape recorder playback, relay links from other spacecraft) to be interleaved with real-time frames.

This 3-bit field enables up to eight "Virtual Channels" to be run concurrently by a particular spacecraft on a particular physical data channel. The sequence in which Virtual Channels are multiplexed is mission-dependent. If only one Virtual Channel is used, these bits shall be set permanently to "000".

(d) Operational Control Field Flag (Bit 15).

This l-bit flag signals the presence (Bit15 = "1") or absence (Bit 15 = "0") of the 32-bit Operational Control Field within the Frame Trailer (see Section 5.5).

5.2.2 Master Channel Frame Count (8 bits). The purpose of this field is to provide a running count of the number of frames which have been transmitted through a single spacecraft physical data channel. Some spacecraft. may have the capability to create more than one physical data channel to the ground, in which case a separate counter will therefore be maintained for each channel. The counter must be long enough to provide a reasonable probability of detecting how many frames were missing if the physical channel is briefly interrupted. The 8-bit field represents a sequential count (modulo 256) of each Transfer Frame generated by the spacecraft on a given physical data channel.

5.2.3 Virtual Channel Frame Count (8 bits). The purpose of this field is to provide individual accountability for each of the eight "Virtual Channels". The 8-bit field represents a sequential count (modulo 256) of the total number of frames which have been transmitted in association with EACH of the virtual channels. It is used in association with the "Virtual Channel ID" field to maintain a separate counter for each of up to eight separate virtual channels.

5.2.4 Frame Data Field Status (16 bits). The purpose of this field is to provide control information necessary to enable Packets or Segments to be extracted from the Frame Data Field. The field is broken into the following subfields:

(a) Secondary Header Flag (Bit 0).

This l-bit flag indicates the presence (Bit 0 = "1") or absence (Bit 0 = "0") of the optional Secondary Header in the Transfer Frame. If present, the Secondary Header shall immediately follow the Primary Header and the beginning of the Frame Data Field shall be correspondingly shifted.

If implemented, the Secondary Header shall appear in every frame transmitted through a physical data channel, and its length shall be fixed within that channel.

(b) Data Field Synchronization Flag (Bit 1).

The normal mode of inserting Packet/Segment data units into the Frame Data Field shall be to synchronously place them on octet boundaries so that they follow directly after each other. The Packets or Segments will thus be permitted to "spill over" into the next frame, and the location of the FIRST Packet/Segment header in a particular frame will be specified by the First Header Pointer field. For this synchronous insertion mode, the Data Field Synchronization Flag shall be set to a "0".

If a Project chooses to place Packet/Segment data ASYNCHRONOUSLY within the frame data field so that octet boundaries are not observed, then the Data Field Synchronization Flag within that frame shall be set to a "1". This situation may occur if the Project elects not to achieve onboard Packet/Segment bit re-justification when a tape recorder is replayed in reverse (thus dumping an unstructured bitstream into the data field), or if the source data do not otherwise conform to the Packet/Segment protocols. IF THIS FLAG IS SET TO A "1", INDICATING ASYNCHRONOUS DATA INSERTION, THE REMAINING FRAME DATA FIELD STATUS INFORMATION MAY NOTBEVALID, AND IT SHALL BE A PROJECT RESPONSIBILITY TO EXTRACT INFORMATION FROM THE FRAME DATA FIELD.

The synchronization status of the Packet/Segment data which are inserted into the Frame Data Field shall be indicated by setting the Data Field Synchronization Flag (Bit 1) as follows:

Bit 1 = 0 : Packet/Segment data are synchronously inserted.
Bit 1 = 1 : Packet/Segment data are asynchronously inserted.

(c) Packet Order Flag (Bit 2).

During normal real-time transmission of information from spacecraft sources to the ground, the Source Packets or Telemetry Segments inserted within the Frame Data Field (as they appear to the ground processor) will be "forward" justified, i.e. they will appear with their most-significant bit transmitted first, and with their sequence counters incrementing in an increasing order. The Packet Order Flag shall be used to indicate certain conditions where the ORDER of the sequence counters within the Packets or Segments may be reversed.

When contact with a ground station is not maintained, or when the transmission capacity of the downlink channel is less than the demands of the data sources, the spacecraft may record telemetry data on an onboard storage device. If this device is a tape recorder, then this Recommendation recognizes that it may be desirable to replay the recorder in a reverse direction, causing the order of the transmitted data to be reversed.

For CCSDS cross-supported missions the baseline requirement is that the spacecraft shall re-justify the BIT DIRECTION of any Packet/Segment data which have been replayed in reverse, prior to inserting them into the Transfer Frame, so that the most significant bit is transmitted first. This will require Packet/Segment synchronization logic at the output of the tape recorder to reverse the bit direction of each Packet/Segment as it is retrieved. Under these conditions, the ORDER of the Packet/Segment sequence counters will be observed to decrease rather than increase.

(Note: if replayed Packets/Segments appear in reverse order, they must be multiplexed together on a separate Virtual Channel: it is not permissable to mix forward-ordered and reverse-ordered Packets/Segments within the same Virtual Channel since the Packet Order Flag would be ambiguous).

The ORDER of the sequence counts within the Packet/Segment data contained within the frame data field shall be indicated by setting the Packet Order Flag (Bit 2) as follows:

Bit 2 = 0 : Packet/Segment sequence count order is "forward".
Bit 2 = 1 : Packet/Segment sequence count order is "reverse".

A discussion of various options for handling tape recorded data is contained in the "Application Notes" (Annex E).

(d) Segment Length Identifier (Bits 3,4).

As discussed in Section 4, Version 2 Telemetry Segments may be implemented within a Virtual Channel as a method of controlling the flow of data from sources which generated very long packets. This 2-bit field identifies the selected maximum length of the standard Version 2 Telemetry Segment, if this option is being used within the Virtual Channel which is formed by the frame. The contents of the field are interpreted as follows:

00 =	256-octet	segment
01 =	512-octet	segment
10 =	1024-octet	segment

If Version 2 Telemetry Segments and Version 1 Source Packets are mixed within the same Virtual Channel, the Version 1 packets may not be longer than the indicated maximum segment length. If the Version 2 Telemetry Segment is not being used within the Virtual Channel, the field shall be set to the "11" state.

(e) First Header Pointer (Bits 5 through 151.

Packet or Segment headers shall be aligned with octet boundaries within the Frame Data Field. The purpose of this field is to point directly to the location of the starting octet of the first Packet or Segment header structure. The location of any subsequent headers within the same Frame Data Field will be determined by a "chaining" procedure whereby the Packet Length field (Section 3.1.3) within each Packet/Segment header structure will be examined to determine where the following header begins.

This ll-bit field contains a binary count "P" (modulo 2048) which, when incremented by "1", points directly to the number of the octet within the Frame Data Field (STARTING AT OCTET # "1", WHICH BEGINS AT THE FIRST BIT OF THE FRAME DATA FIELD) that contains the first octet of the first Packet or Segment header structure. The count "P" is expressed as follows:

P ={ (Number of the octet) - 1 }

If the frame DOES NOT contain a Transfer Frame Secondary Header, the first bit of Octet # 1 within the Frame Data Field occurs immediately after the last bit of the Primary Header.

If the frame DOES contain a Secondary Header, then the first bit of Octet # 1 within the Frame Data Field occurs immediately after the last bit of the Secondary Header, i.e. it is offset by the length of the Secondary Header, which is specified within the Secondary Header Identification field. The value of the First Header Pointer is not affected by the existence of a Secondary Header.

If no Packet or Segment header structure starts in the Data Field, the First Header Pointer shall be set to ~11111111111" ("all ones") This situation may occur if a long packet is segmented using the Virtual Channel technique.

If a Virtual Channel does not contain any valid Packet or Segment data, the First Header Pointer shall be set to "11111111110" ("all ones minus one"). This may be used to signal an "Idle Channel" if no real or fill packets are available for transmission within the frame.

Since Packets or Segments may begin at any point within the Frame Data Field, it is possible that a Packet/Segment header may be split between successive frames. The rules for handling this situation are as follows:

(i) If the FIRST Packet/Segment header starts at the end of the Data Field within frame (N) and spills-over into frame (N+1), the First Header Pointer in frame (N) shall indicate the start of this Packet/Segment header.

(ii) If ANY Packet/Segment header is split between frames (N) and (N+1), the pointer in frame (N+1) shall ignore the residue of the split header and shall only indicate the start of any subsequent new Packet/Segment header within frame (N+1).

5.3 TRANSFER FRAME SECONDARY HEADER (Optional).

The Transfer Frame Secondary Header is optional: its presence or absence is indicated by the Secondary Header Flag within the Primary Header. If implemented, the Secondary Header must be of fixed length and must appear in every frame transmitted through a physical data channel. Every Secondary Header shall begin with a single octet containing the "Secondary Header Identification", and shall have the general format indicated in Figure 5-2.

5.3.1 Secondary Header Identification (8-bits). This field defines the version and length of the Secondary Header: it is mandatory if a Secondary Header is present. It is separated into two subfields:

(a) Secondary Header Version Number (Bits 0,1).

This 2-bit subfield shall indicate which of up to four Secondary Header versions is being used. By changing the Version Number, future variations of the Secondary Header structure become possible. At present, only Version 1 (Bits 0,1 = "00") is recognized, and all other versions are reserved for future application. Within Version 1, the remainder of the Secondary Header format is defined as follows:

(b) Secondary Header Length (Bits 2 through 7).

This 6-bit subfield shall contain a binary count "S" of the total number of octets contained within the entire Transfer Frame Secondary Header (including the Secondary Header Identification field itself). The count "S"is expressed as follows:

$$s = \{ (\text{Total number of octets}) - 1) \}$$

When a Secondary Header is present, this count consequently may be used to compute the number of octet boundaries by which the first bit of the Frame Data Field is offset from the last bit of the Primary Header.

5.3.2 Secondary Header Data (n x 8-bits). For Version 1 of the Secondary Header, this field (which must contain an integral number of octets and can be up to 63-octets long) contains a "Real Time Data Insert' of information that is required for various spacecraft monitoring and control applications. Guidelines for implementing the Real Time Data Insert are presented in the "Application Notes" (Annex E).

5.4 TRANSFER FRAME DATA FIELD

This field, which must exist as an integer number of octets, contains user application data (e.g. Packets or Segments) to be transferred from the spacecraft to the ground.

Packets or Segments shall be inserted contiguously into the Data Field on octet boundaries, with the location of the octet containing the first header being indicated by the First Header Pointer in the frame header. Subsequent headers are located by examining the "length" field in each packet, or by counting forward by the fixed Telemetry Segment length, whichever is applicable.

When there are no data available for transmission from any source, an "Idle Packet" may be inserted by the spacecraft data system for the purpose of keeping the frame running synchronously. The Idle Packet shall have the format of a Version 1 Source Packet or a Version 2 Telemetry Segment, its Application Process ID shall read "all ones", and its length may equal the minimum-available packet size.

The maximum length of the Frame Data Field depends on whether the optional Transfer Frame Secondary Header and Transfer Frame Trailer fields are present, and on the length of the Synchronization Marker. As discussed in Reference [2], if frame lengths shorter than the 8920-bit maximum are implemented and the frame is encoded using the recommended Reed-Solomon algorithm, then the length of the Frame Data Field must be selected bearing in mind the constraint that "Virtual Fill" must occur in fixed increments.

5.5 TRANSFER FRAME TRAILER (Optional).

The Transfer Frame Trailer provides a mechanism for inserting the following optional information into the trailing octets of the frame:

(a) An Operational Control Field, which facilitates closed-loop reporting of certain standardized real-time activities.

(b) A Frame Error Control Word, which facilitates detection of errors which may have occurred within the frame.

5.5.1 Operational Control Field (32 bits) (Optional). The purpose of this field is to provide a standardized mechanism for reporting a small number of real-time functions (e.g. telecommand verification or spacecraft clock calibration). The leading bit of the field (Bit 0) is a 'Type" flag that indicates which function is being reported.

A "Type 1" report (Bit 0 = "0") contains a "Command Link Control Word" which is used to provide acceptance reporting for spacecraft that are compatible with the Transfer Frame layer of the Packet Telecommand concept: the internal format of the Command Link Control Word is fully defined in Reference [3]. The format of a "Type 2' report (Bit 0 = "1") is currently undefined and is reserved for future application.

The presence or absence of this field is signalled by the Operational Control Field Flag within the Transfer Frame Primary Header. If present, the field must occur within EVERY frame transmitted through a physical data channel.

If the optional Frame Error Control Word is NOT present, the Operational Control Field occupies the four trailing octets of the Transfer Frame: if the Error Control Word IS present, the field is displaced towards the beginning of the frame by two octets.

5.5.2 Frame Error Control Word (16-bits) (Optional). The purpose of this 16-bit field is to provide a capability for detecting errors which may have been introduced into the frame during the data handling processes.

If the Transfer Frame is NOT Reed-Solomon encoded, the presence of the Frame Error Control Word is mandatory. The field is optional if the Frame is synchronously contained within the data space of a Reed-Solomon codeblock. If present, the field occupies the two trailing octets of the Transfer Frame. Presence or absence of the field is implied from the Spacecraft Identifier and is specified to the ground network as a mission set-up parameter.

Guidelines for generating the Frame Error Control Word are presented in Annex A.

ANNEX A:

TRANSFER FRAME ERROR DETECTION ENCODING/DECODING GUIDELINE

(THIS ANNEX IS NOT PART OF THE RECOMMENDATION)

Purpose:

This Annex provides a description of the recommended error detection encoding/decoding procedure which is to be used within the Transfer Frame.

Status:

Final selection of an encoding polynomial to generate the 16-bit Frame Error Control Word is currently under study. Meanwhile, a recommended method is to encode the entire Transfer Frame (starting with the first bit of the Synchronization Marker and ending with the last bit of the Frame Trailer) into a binary cyclic block code using the HDLC/ADCCP generating polynomial described in this Annex.

A-1. ENCODING AND DECODING OF TRANSFER FRAMES USING THE ADVANCED DATA COMMUNICATION CONTROL PROCEDURE (ADCCP) FRAME CHECK SEQUENCE (PCS) PROCEDURES

This Annex describes the error detection encoding/decoding procedure that is recommended for Transfer Frame coding.

Parity encoding is mandatory if the Transfer Frame is NOT Reed-Solomon encoded prior to transmission. If error detection coding of the Transfer Frame is performed, it is recommended that it should be compatible with the procedures described in this Annex. This does not preclude the use of a more powerful error detection code than that described herein provided that all codewords of the more powerful code are also codewords of the Advanced Data Communications Control Procedure (ADCCP) code. This implies that the generating polynomial of the more powerful code must be divisible by the ADCCP generating polynomial and that the more powerful code. adopts the presetting procedures to be described.

Commercially available integrated circuits can be purchased to perform the encoding/decoding functions. If these modules are used it may not be necessary for the user to be concerned with the details of the coding procedure.

The ADCCP code has the following capabilities when applied to an encoded block of less than 32,768 (2^{15}) bits:

(1) All error sequences composed of an odd number of bit errors will be detected.

(2) All error sequences containing two-bit errors anywhere in the encoded block will be detected.

(3) If a random error sequence containing an even number of bit errors (greater than or equal to 4) occurs within the block, the probability that the error will be undetected is approximately 2^{-15} (or $\simeq$ 3 x 10^{-5}).

(4) All single error bursts spanning 16 bits or less will be detected provided no other errors occur within the block.

(5) This code is only intended for error detection purposes and no attempt should be made to utilize it for error correction, since it would falsely decode some blocks containing double errors.

A-l.1 ENCODING PROCEDURE

The encoding procedure accepts an (n-16)-bit data block and generates a systematic binary (n,n-16) block code by appending a 16-bit Frame Check Sequence (FCS) as the final 16 bits of the block. This FCS is inserted into the Frame Error Control Word of the Transfer Frame Trailer. The equation for the FCS is:

$$FCS = [X^{16} . H(X) \oplus X^{n} . L(X)] \text{ modulo } G(X)$$

where: M(X) is the (n-16)-bit message to be encoded expressed as a polynomial with binary coefficients

L(X) is the presetting polynomial given by:

$$L(X) = \sum_{i=0}^{15} xi \quad \text{(all "1" polynomial of order 15)}$$

G(X) is the CCITT Recommendation V.41 generating polynomial given by:

$$G(X) = X^{16} + X^{12} + X^{5} + 1$$

n is the number of bits in the encoded message

$\oplus$ is the modulo 2 addition operator (Exclusive OR)

Note that the encoding procedure differs from that of a conventional cyclic block encoding operation in that:

(1) The $X^{n}.L(X)$ factor has the effect of presetting the shift register to an all "1" state (rather than a conventional all "0" state) prior to encoding.

A-1.2 DECODING PROCEDURE

The error detection syndrome, S(X), is given by

$$S(X) = [C^{*} (X) \oplus X^{n} . L(X)] \text{ MODULO } G(X)$$

where C*(X) is the received block in polynomial form and,

S(X) is the syndrome polynomial which will be zero if no error is detected and non-zero if an error has been detected.

ANNEX B:

TRANSFER FRAME SYNCHRONIZATION MARKER

(THIS ANNEX **IS NOT** PART OF THE RECOMMENDATION)

Purpose:

This Annex specifies the bit pattern to be used for the Attached Synchronization Marker when the Transfer Frames are NOT Reed-Solomon encoded.

Status:

The exact pattern of the 32-bit marker is currently under study.

ANNEX C:

SOURCE PACKET SECONDARY HEADER FORMAT GUIDELINE

(THIS ANNEX IS NOT PART OF THE RECOMMENDATION)

Purpose:

This Annex provides preliminary guidelines for the structure of the Secondary Header containing the results of onboard application processes which aid the interpretation of the source data set encapsulated within a Version 1 Source Packet.

Status:

This Annex is currently under development by the CCSDS.

C-l OVERVIEW.

The Secondary Header provides the capability for insertion of ancillary and correlative data which increase the processing autonomy of the Source Data contained within a packet. Candidate information for inclusion within the Secondary Header includes measurement time, spacecraft attitude, instrument pointing information, etc. The CCSDS has not yet formulated a comprehensive guideline for the format of the Source Packet Secondary Header; therefore, this Annex only contains preliminary information.

One important parameter which may be needed in the Secondary Header is a time-tag which, when concatenated with the Source Sequence Count, provides unique identification of the packet. This time-tag must be included if the user requires that the ground network provides long-term packet management and accounting services (archiving, correlation with other data sets, etc.). Since these long-term services have potential cross-support significance, the CCSDS has made the following provisional recommendations with respect to Secondary Headers:

(i) The presence of a Secondary Header is optional, and may be included at the discretion of the local user or Project organization. The presence or absence of a Secondary Header is signalled by the Secondary Header Flag within the packet header. As a minimum requirement, the Secondary Header shall contain a timecode which references the packet to a known epoch.

(ii) The timecode which is selected for use within the Source Packet Secondary Header shall conform to the CCSDS Recommendation for Timecode Formats, Reference [4].

c-2 CONTENTS OF THE SECONDARY HEADER.

When a Secondary Header is present within a Source Packet, it shall be structured according to Figure C-l.

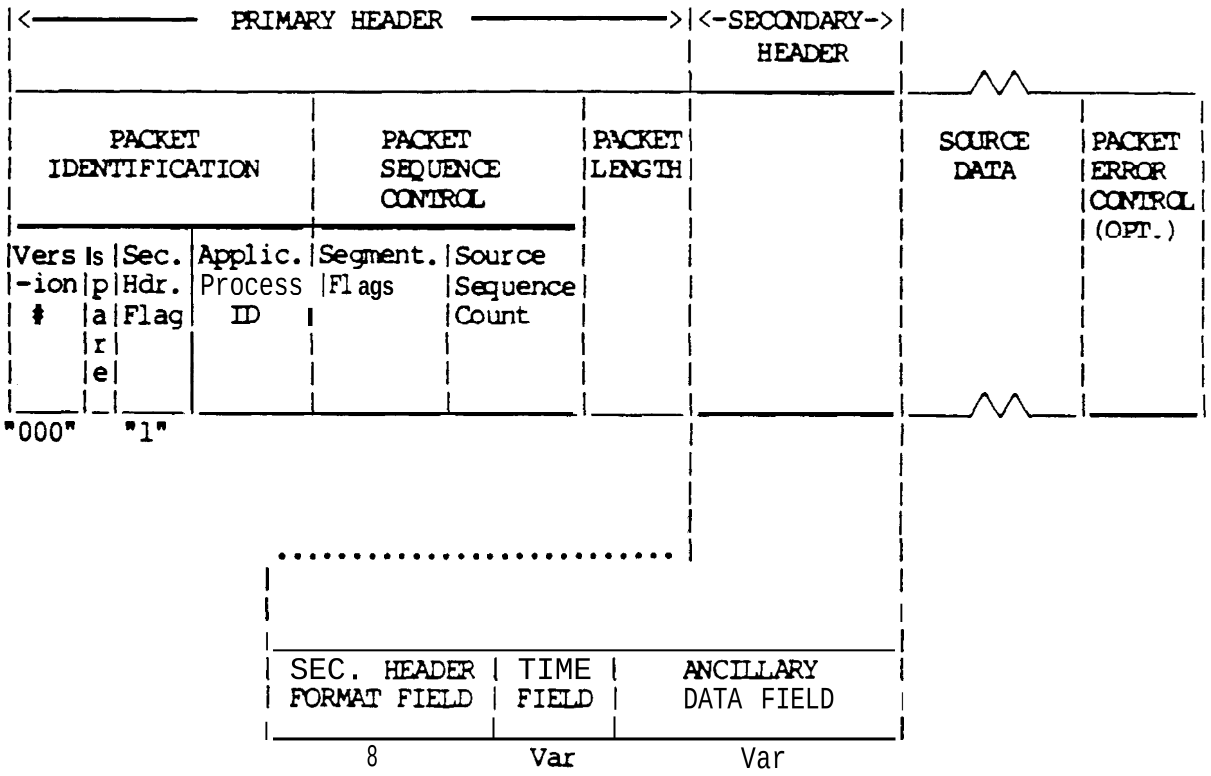


FIGURE C-l: SOURCE PACKET SECONDARY HEADER STRUCTURE

C-2.1 SECONDARY HEADER FORMAT FIELD

The internal format of this 8-bit field is currently under study by the CCSDS.

C-2.2 TIME FIELD

The format of this field is currently under study by the CCSDS. The contents of the field shall be compatible with the CCSDS Timecode Formats specification, Reference [4].

C-2.3 ANCILLARY DATA FIELD (optional)

The internal format of this field is currently under study by the CCSDS.

ANNEX D:

SOURCE PACKET ERROR DETECTION ENCODING/DECODING GUIDELINE

(THIS ANNEX **IS NOT** PART OF THE RECOMMENDATION)

Purpose:

This Annex provides a description of the recommended error detection encoding/decoding procedure which is to be used within the Source Packet.

Status:

This Annex is currently under development by the CCSDS. As an interim measure until this Annex is supplied, it is suggested that the specification contained in Annex-A may be used for Source Packet error control.

ANNEX E:

"APPLICATION NOTES" FOR PACKET TELEMETRY

(THIS ANNEX **IS NOT** PART OF THE RECOMMENDATION)

Purpose:

The Packet Telemetry formats defined in this Recommendation are layered so that various levels of interface compatibility are possible. This Annex describes how a Project may implement complete or partial compatibility with Packet Telemetry.

Status:

This Annex is currently under development by the CCSDS, and is therefore incomplete. The following items are included for information.

1. Section 5.2.4 (c): Packet Order Flag.

Three recognized options exist for implementing reverse tape recording modes:

(i) The complete telemetry stream may be recorded as a series of Telemetry Frames. This entire stream may then later be replayed in reverse direction and dumped to the ground OVER A PHYSICAL DATA CHANNEL WHICH IS SEPARATE FROM THAT USED TO TRANSMIT REAL-TIME DATA. In this case, the Packet Order Flag shall indicate the status of the Packets or Segments when the frames were originally recorded.

(ii) The complete telemetry stream may be recorded as a series of Telemetry Frames, each having their Packet Order Flag set as appropriate during recording. This entire stream may then later be replayed in reverse direction as a pure bitstream for insertion within the Data Field of new frames which form a separate playback Virtual Channel. These playback frames may then be interleaved with other frames which form Virtual Channels that contain real-time Packets or Segments. In this case the replayed bitstream will be inserted into the playback Virtual Channel asynchronously, with the "Data Field Synchronization Flag" for this channel set to a "1" and the Packet Order Flag consequently ignored. (Note: precautions must be taken to ensure that the replayed synchronization marker occurring periodically within the frame data field does not interfere with the overall frame synchronization strategy. As an example, the reverse-justified synchronization marker should be distinguishable from the forward-justified pattern.)

(iii) Packets or Segments may be recorded with or without first encapsulating them within Transfer Frames. These Packets or Segments may later be replayed in reverse direction, and re-synchronized onboard the spacecraft for normal insertion into the Data Field of new real-time Transfer Frames.

2. Section 5.3.2: Real Time Data Insert.

The format, utilization and operational procedures associated with the Real Time Data Insert field are all mission-dependent and shall be the subject of detailed cross-support agreements between the Agencies involved.

ANNEX F:

RATIONALE FOR PACKET TELEMETRY

(THIS ANNEX IS NOT PART OF THE RECOMMENDATION)

Purpose:

This Annex presents some of the rationale which led to the evolution of the Packet Telemetry concept.

Status:

This Annex is complete.

F-l PACKET TELEMETRY RATIONALE

Packet Telemetry represents an evolution-ary step from the traditional Time-Division Multiplex (TDM) method of transmitting scientific, applications, and engineering data from spacecraft sources to users on the ground. The Packet Telemetry process conceptually involves:

(1) Encapsulating, at the source, experiment observational data together with the ancillary data used to subsequently interpret the observational data, thus forming an autonomous "packet" of information in real time on the spacecraft.

(2) Providing a standardized mechanism whereby autonomous packets from multiple data sources on the spacecraft can be inserted into a common "frame" structure for transfer to'the ground through noisy data channels, and delivered to facilities-where the packets may be extracted for delivery to the user.

The packet telemetry process has the conceptual attributes of:

(1) Facilitating the acquisition and transmission of instrument data at a rate appropriate for the phenomenon being observed.

(2) Defining a logical interface and protocol between an instrument and its associated ground support equipment which remains constant throughout the life-cycle of the instrument (bench test, integration, flight, and possible re-use).

(3) Simplifying overall system design by allowing a microprocessor-based symmetric design of the instrument control and data paths ("command packets in, telemetry packets out") compatible with commercially available components and interconnection protocol standards.

(4) Eliminating the need for mission-dependent hardware and/or software at intermediate points within the distribution networks through which space data flows: in particular, enabling the multimission components of these networks to be designed and operated in a highly automated fashion, with consequent cost and performance advantages.

(5) Facilitating interoperability of spacecraft whose telemetry interfaces conform to these guidelines, i.e., allowing very simple cross-strapping of spacecraft and ground network capabilities between Agencies.

(6) Enabling the delivery of high-quality data products to the user community in a mode which is faster and cheaper than would be possible with conventional telemetry.

ANNEX G:

SUMMARY OF SEGMENTATION OPTIONS

(THIS ANNEX IS NOT PART OF THE RECOMMENDATION)

Purpose:

This Annex provides a summary of the various options which exist for segmenting very long Source Packets, in order to achieve flow control through the space-to-ground data channel.

Status:

This Annex is complete.

G-1. SEGMENTATION SUMMARY.

Several options for segmenting long Source Packets are specified in Sections 4 and 5 of the Recommendation for Packet Telemetry. In selecting the segmentation method to be used for a particular mission, the following system considerations may be important:

(a) Segmentation should not introduce extra overhead into short packets which have no need to be segmented.

(b) It should be possible to mix short unsegmented packets on the same virtual channel with long source packets which have been divided into segments.

(c) It is highly desirable to implement a solution which uses a single protocol for both segmented and unsegmented packets, in which data fields are interpreted in singular, consistent ways.

(d) For a given mission, a fixed maximum segment length should be selected. When long packets are broken into segments, the segment lengths may be equal to the mission-fixed maximum, except for the last segment which may contain the residue of the original packet.

(e) The segmentation solution should involve the simplest possible algorithms for extracting the packets and segments from the Transfer Frame, and for reconstituting the packets, since these algorithms must operate at full incoming telemetry bit rate.

Table G-l presents a summary of the major attributes of the various alternative methods.

OPTION:	PRINCIPLE:	SEGMENTS FORMED BY:	GROUND PROCESSOR USES:•	USER RECEIVES:	OVERHEAD:
Source - internal using Version-1 Source Packet (4.1)	Use Source Sequence Counter to identify each segment within the Packet	Source	Packet Sequence Count and Length fields to extract segments.	Segments	6-octets/segment + 4-octets (length & count) in the first segment
Spacecraft segmentation using Version- Source Packet (4.2)	Nest the user Source Packet within an "outer" spacecraft Source Packet	Spacecraft data system	Outer spacecraft Source Packet to extract and recombine segments.	Source Packet	6-octets/segment + 6-octets (original header) in the first segment
Spacecraft segmentation using Version-2 Telemetry Segment (4.3)	Source Packet length field decrenented by fixed binary Segment Length	Spacecraft data system	Known fixed length to extract each segment; inferred sequence to recombine them.	Source Packet	6-octets/segment.
Virtual Channelization (5.2.2b)	Long Packets assigned to their own dedicated Transfer Frame	Spacecraft data system	Virtual Channel ID in Transfer Frame header	Source Packet	Depends on design: may be zero.

TABLE G-1: SUMMARY OF SEGMENTATION OPTIONS

CONSULTATIVE
COMMITTEE FOR
SPACE DATA SYSTEMS

RECOMMENDATION FOR SPACE
DATA SYSTEM STANDARDS:

TELEMETRY CHANNEL CODING

"BLUE BOOK"

MAY 1984

CCSDS

DEDICATION

This document is dedicated to the memory of Mr. Michel F. Pellet of the European Space Agency. His vision and leadership were an inspiration to many technical personnel who contributed to this Recommendation. He will be deeply missed by many members of the CCSDS, and by all people who value the increased international understanding which is achieved through cooperation in the peaceful scientific exploration of space.

AUTHORITY

Issue:	Blue Book, Issue-l.
Date:	May 1984.
Location:	CCSDS Plenary Meeting, May, 1984, Tokyo, Japan.

This Recommendation reflects the oonsensus technical agreement of the following member Agencies of the Consultative Committee for Space Data Systems (CCSDS):

- Centre National D'Etudes Spatiales (CNES)/France
- Deutsche Forschungs-u. Versuchsanstalt fuer Luft und Raumfahrt e.V (DFVLR)/West Germany
- European Space Agency (ESA)/Europe.
- Indian Space Research Organization (ISRO)/India
- Instituto de Pesquisas Espaciais (INPE)/Brazil
- National Aeronautics and Space Administration (NASA)/USA
- National Space Development Agency of Japan (NASDA)/Japan.

The following observer Agencies also concur with this Recommendation:

- Department of Communications, Communications Research Centre (DOC-CRC)/Canada
- Institute of Space and Astronautical Science (ISAS)/Japan
- Radio Research Laboratory (RRL)/Japan

This Recommendation is published and maintained by:

CCSDS Secretariat
Communications and Data Systems Division
Code-TS
National Aeronautics and Space Administration
Washington
DC 20546
USA

STATEMENT OF INTENT

The Consultative Committee for Space Data Systems (CCSDS) is an organization officially established by the management of seven member space Agencies. The Committee meets periodically to address data systems problems that are common to all participants, and to formulate sound technical solutions to these problems. Inasmuch as participation in the CCSDS is completely voluntary, the results of Committee actions are termed RECOMMENDATIONS and are not considered binding on any Agency.

This RECOMMENDATION is issued by, and represents the consensus of, the CCSDS Plenary body. Agency endorsement of this RECOMMENDATION is entirely voluntary. Endorsement, however, indicates the following understandings:

- Whenever an Agency establishes a CCSDS-related STANDARD, this STANDARD will be in accord with the relevant RECOMMENDATION. Establishing such a STANDARD does not preclude other provisions which an Agency may develop.

- Whenever an Agency establishes a CCSDS-related STANDARD, the Agency will provide other CCSDS member Agencies with the following information:

  - The STANDARD itself.

  - The anticipated date of initial operational capability.

  - The-anticipated duration of operational service.

- Specific service arrangements shall be made via memoranda of agreement. Neither this RECOMMENDATION nor any ensuing STANDARD is a substitute for a memorandum of agreement.

No later than five years from its date of issuance, this Recommendation will be reviewed by the CCSDS to determine whether it should: (1) remain in effect without change: (2) be changed to reflect the impact of new technologies, new requirements, or new directions; or, (3) be retired or cancelled.

FOREWORD

This document is a technical Recommendation for use in developing telemetry channel coding systems and has been prepared by the Consultative Committee for Space Data Systems (CCSDS). The telemetry channel coding concept described herein is the baseline concept for spacecraft-to-ground data communication within missions that are cross-supported between Agencies of the CCSDS.

This Recommendation establishes a common framework and provides a common basis for the coding schemes used on spacecraft telemetry streams. It allows implementing organizations within each Agency to proceed coherently with the development of compatible derived Standards for the flight and ground systems that are within their cognizance. Derived Agency Standards may implement only a subset of the optional features allowed by the Recommendation and may incorporate features not addressed by the Recommendation.

Through the process of normal evolution, it is expected that expansion, deletion or modification to this document may occur. This Recommendation is therefore subject to CCSDS document management and change control procedures which are defined in Reference [1].

DOCUMENT CONTROL

Issue	Title	Date	Status/Remarks
1	Recommendation for Space Data Systems Standards: Telemetry Channel Coding.	May 1984	Original Issue

CONTENTS

Sections Page

Figures

Annexes

REFERENCES

[1] "Procedures Manual for the Consultative Committee for Space Data Systems,' CCSDS, May 1984 or later issue.

[2] "Telemetry Coding Standard,. European Space Agency ESA:TTC-A-03, Issue No. 1, December 1979.

[3] "Tracking and Data Relay Satellite System (TDRSS) Users' Guide," NASA-Goddard Space Flight Center STDN 101.2, Rev. 4, January, 1980.

[4] "Deep Space Network/Flight Project Interface Design Book, Volume II: Proposed DSN Capabilities,. NASA-Jet Propulsion Laboratory, Document 810-5, Rev. D.

[5] "Recommendation for Space Data System Standards: Packet Telemetry,' CCSDS, Issue-l, May 1984.

[6] Miller, R. L., et al, 'On the Error Statistics of Viterbi Decoding and the Performance of Concatenated Codes,. NASA-Jet Propulsion Laboratory Publication 81-9, September 1, 1981.

[7] Perlman, M., and J. Lee, 'Reed-Solomon Encoders - Conventional vs. Berlekamp's Architecture,. NASA-Jet Propulsion Laboratory Publication 82-71, December 1, 1982.

[8] Odenwalder, J. P., et al, "Hybrid Coding Systems Study," Final Report, Contract NAS2-6722, Linkabit Corp., San Diego, Sept. 1972.

1 INTRODUCTION

1.1 PURPOSE

The purpose of this document is to establish a common Recommendation for space telemetry channel coding systems to provide cross-support among missions and facilities of member Agencies of the Consultative Committee for Space Data Systems (CCSDS.) In addition, it provides focussing for the development of multi-mission support capabilities within the respective agencies to eliminate the need for arbitrary, unique capabilities for each mission.

Telemetry channel coding is a method by which data can be sent from a source to a destination by processing data so that distinct messages are created which are easily distinguishable from one another. This allows reconstruction of the data with low error probability, thus improving the performance of the channel.

This document was prepared by the CCSDS primarily for the purpose of facilitating the cross-support concept through standardizing key items of data systems compatibility. While the CCSDS has no power of enforcement, it is expected that this recommendation will be incorporated into each respective Agency's data systems standards, and through them, will apply to all missions that wish to utilize telemetry channel coding for cross-support.

1.2 SCOPE

Several space telemetry channel coding schemes are described in this document. The characteristics of the codes are specified only to the extent necessary to ensure interoperability and cross-support. The specification does not attempt to quantify the relative coding gain or the merits of each approach discussed, nor the design requirements for encoders or decoders. Some performance information is included in Annex A, Rationale.

This recommendation does not require that coding be used on ail cross supported missions. However, for those planning to use coding, the recommended codes to be used are those described in this document.

The rate 1/2 convolutional code recommended for cross-support is described in Section 2, "Convolutional Coding." Depending on performance requirements, this code alone may be satisfactory.

Users of the NASA Tracking and Data Relay Satellite System (TDRSS) may be required to use periodic convolutional interleaving in addition to the convolutional code above. This approach is described in Section 3, "Convolutional Coding with Interleaving for Tracking and Data Relay Satellite Operations."

Where a greater coding gain is needed than can be provided by the convolutional code alone, a standard Reed-Solomon outer code may be concatenated for improved performance. The specification of the Reed-Solomon code selected for cross-support is given in Section 4, "Reed-Solomon Coding." It should be noted that if a spacecraft, utilizing the services of TDRSS, incorporates Reed-Solomon coding, it is the responsibility of the user project to provide the required Reed-Solomon decoding.

1.3 APPLICABILITY.

This Recommendation applies to telemetry channel coding applications of space missions anticipating cross support among CCSDS member Agencies at the coding layer. In addition, it serves as a guideline for the development of compatible internal Agency Standards in this field, based on good engineering practice.

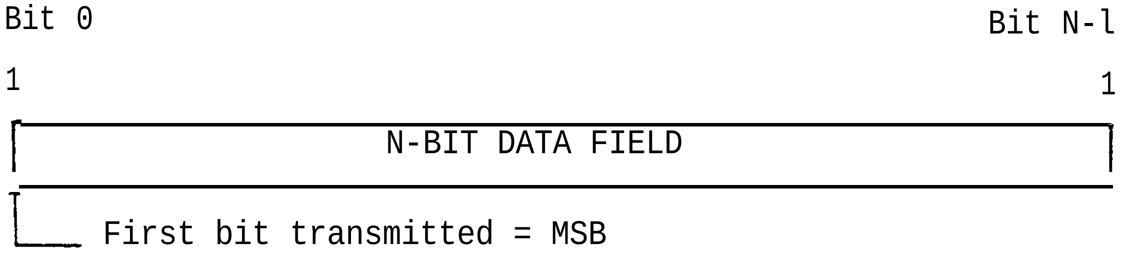
1.4 BIT NUMBERING CONVENTION AND NOMENCLATURE

The following "Caution" should be observed when interpreting the bit numbering convention which is used throughout this CCSDS Recommendation:

CAUTION

In this document, the following convention is used to identify each bit in a forward-justified N-bit field.

The first bit in the field to be transmitted (i.e. the most left justified when drawing a figure) is defined to be "Bit 0"; the following bit is defined to be "Bit 1" and so on up to "Bit N-l". When the field is used to express a binary value (such as a counter), the Most Significant Bit (MSB) shall be the first transmitted bit of the field, i.e. "Bit 0".



In accordance with modern data communications practice, spacecraft data fields are often grouped into 8-bit "words" which conform to the above convention. Throughout this Recommendation, the following nomenclature is used to describe this grouping:

"8-bit word" = "Octet"

2 CONVOLUTIONAL CODING

The basic code selected for cross-support is a rate 1/2, constraint-length 7 convolutional code. It may be used alone, as described in this section, or in conjunction with enhancements described in the following sections. While slightly different conventions of this code, currently in use by some member Agencies, may continue to be supported for an interim period, it is the recommendation of the CCSDS to universally adopt the single convention described herein.

This recommendation is a non-systematic code and a specific decoding procedure, with the following characteristics:[1,2]

a.	Nomenclature:	Convolutional code with maximum-likelihood (Viterbi) decoding.
b.	Code rate:	1/2 bit per symbol
c.	Constraint length:	7 bits
d.	Connection vectors:	G1 = 1111001; G2 = 1011011
e.	Phase relationship:	G1 is associated with first Symbol
f.	Symbol inversion:	On output path of G2

An encoder block diagram with the recommended convention is shown in Fig. 2-l.

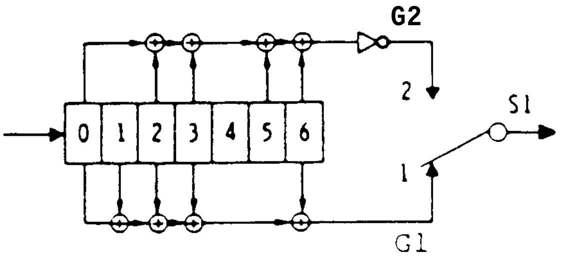
It is recommended that soft bit decisions with at least 3-bit quantization be used whenever constraints (such as location of decoder) permit.

Footnote 1: The following upper bounds to the data rates for spacecraft telemetry reception may exist because of symbol synchronizer or decoder limitations:

(a) ESA: Maximum symbol rate is obtained at a data rate of 1 Mb/s NRZ-L, or 500 kb/s split-phase. (Ref. [2])

(b) NASA-GSFC: Maximum symbol rate is obtained at a data rate of 3 Mb/s NRZ. (Ref. [3])

(c) NASA-JPL: Current maximum symbol rate is 250 ksymbols/s. A planned upgrade will be decoder-limited at 250 kb/s. (Ref. [4])



NOTES :

1. □ = Single bit delay

2. For every input bit, two symbols are generated by completion of a cycle for S1: Pos'n 1, Pos'n 2.

3. S1 is in the position shown (1) for the first symbol associated with an incoming bit.

4. ⊕ = modulo-2 adder.

5. —▷○— = inverter.

Figure 2-l. Convolutional Encoder Block Diagram.

Footnote 2: When suppressed-carrier modulation systems are used, NRZ-M or NRZ-L may be used as a modulating waveform. If-the user contemplates conversion of his modulating waveform from NRZ-L to NRZ-M, such conversion should be performed on-board at the input to the convolutional encoder. Correspondingly, the conversion on the ground from NRZ-M to NRZ-L should be performed at the output of the convolutional decoder. This avoids unnecessary link performance loss and is shown in Fig. A-l.

CAUTION: When a fixed pattern in the symbol stream is used to provide node synchronization for the Viterbi decoder, care must be taken to account for any translation of the pattern due to the modulating waveform conversion.

3 CONVOLUTIONAL CODING WITH INTERLEAVING FOR TRACKING AND DATA RELAY SATELLITE OPERATIONS

3.1 INTRODUCTION

Users of the TDRSS S-band Single Access (SSA) Channel, where the channel symbol rate exceeds 300 ks/s, will be required to employ interleaving in conjunction with the convolutional code which has been described in Section 2. Users are cautioned that if such interleaving is not used under these conditions, the Goddard Space Flight Center Networks Directorate does not guarantee the specified performance and will not be obligated to troubleshoot the system in case of problems. (Ref. [3])

It should be noted that this interleaving is totally separate and distinct from the interleaving used in conjunction with the Reed-Solomon code described in Section 4.

3.2 DESCRIPTION

The type of interleaving required is called "Periodic Convolutional Interleaving" and is specified in Appendix J of the TDRSS Users's Guide. (Ref. [3])

3.3 BYPASS CAPABILITY

A TDRSS-compatible spacecraft using the Periodic Convolutional Interleaving specified in this section must be capable of bypassing its Periodic Convolutional Interleaver in the event direct support from a non-TDRSS ground tracking station is desired. This is because the interference that this interleaving is designed to protect against is not harmful in this configuration, and moreover, the necessary de-interleavers do not exist at these ground stations.

4 REED-SOLOMON CODING

4.1 INTRODUCTION

While a convolutional code provides good forward error correction capability in a gaussian noise channel, significant additional improvement (particularly to correct bursts of errors from the Viterbi decoder) can be obtained by concatenating a Reed-Solomon (R-S) code with the convolutional code. The Reed-Solomon code forms the outer code, while the convolutional code is the inner code. The overhead associated with the Reed-Solomon code is comparatively low, and the improvement in the error performance can often provide the nearly-error-free channel required to support efficient automated ground handling of space mission telemetry.

The user is cautioned that the R-S outer code described in this section is not intended for use except when concatenated with the inner convolutional code described in Section 2.

The TDRS System does not furnish any Reed-Solomon decoding services.

4.2 SPECIFICATION

The parameters of the selected Reed-Solomon code are as follows:

(a) J = 8 bits per R-S symbol

(b) E = 16 R-S symbol error correction capability within a Reed-Solomon codeword.

(c) General characteristics of Reed-Solomon codes

1. J, E, and I, the depth of interleaving, are independent parameters.

2. $n = 2^J - 1 = 255$ symbols per R-S codeword.

3. 2E is the number of R-S symbols among n symbols of an R-S codeword representing checks.

4. k= n-2E is the number of R-S symbols among n R-S symbols of an R-S codeword representing information.

(d) Field generator polynomial:

$$F(x) = x^8 + x^7 + x^2 + x + 1$$

over GF(2)

e) Code generator polynomial:

$$g(x) = \prod_{j=112}^{143} (x - \overline{a^{11j}}) = \sum_{i=0}^{32} G_i x^i$$

over $GF(2^8)$,
where $F(a) = 0$.

It should be recognized that a11 is a primitive element in $GF(2^8)$ and that F(x) and g(x) characterize a (255,223) Reed-Solomon code.

(f) The selected code is a systematic code. This results in a systematic codeblock.

(g) Symbol Interleaving

Symbol interleaving is accomplished in a manner functionally described with the aid of Fig. 4-l. (It should be noted that this functional description does not necessarily correspond to the physical implementation of an encoder.)

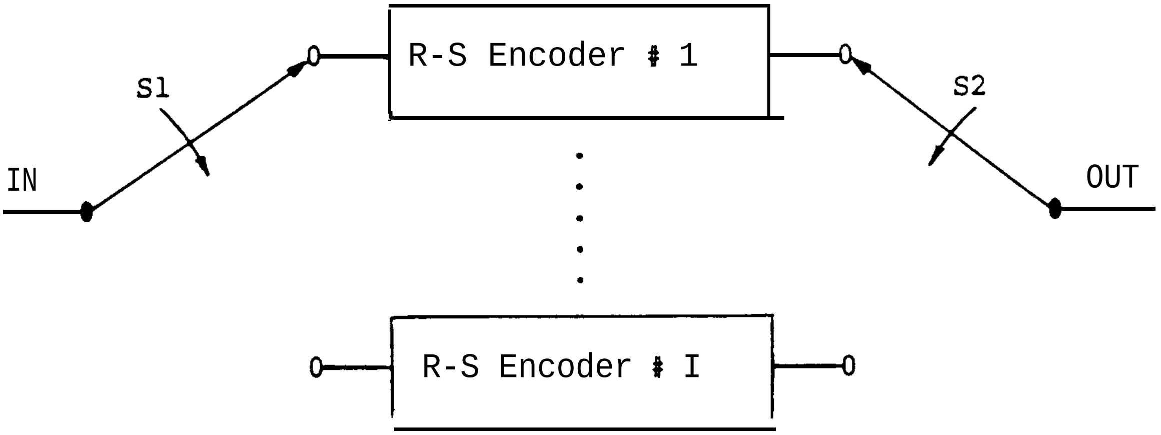


Fig. 4-l. Functional Representation of R-S Interleaving

Data bits to be encoded into a single Reed-Solomon Codeblock enter at the port labeled "IN." Switches S1 and S2 are synchronized together and advance from encoder to encoder in the sequence 1,2, . . . I, 1 . . . , spending one R-S symbol time (8 bits) in each position.

One codeblock will be formed from 2231 R-S symbols entering "IN." In this functional representation, a space of 321 R-S symbols'in duration is required between each entering set of 2231 R-S information symbols.

Due to the action of S1, each encoder accepts 223 of these symbols, each symbol spaced I symbols apart (in the original stream.) These 223 symbols are passed directly to the output of each encoder. The synchronized action of S2 reassembles the symbols at the port labeled "OUT" in the same way as they entered at "IN."

Following this, each encoder outputs its 32 check symbols, one symbol at a time, as it is sampled in sequence by S2.

If, for I=5, the original symbol stream is

$$d^1_1 \ldots d^5_1 d^1_2 \ldots d^5_2 \ldots\ldots d^1_{223} \ldots d^5_{223} \qquad [32 \text{ x } 5 \text{ spaces}]$$

then the output is the same sequence followed by the [32x5] parity symbols as shown below:

$$P^1_1 \ldots P^5_1 \ldots\ldots P^1_{32} \ldots P^5_{32}$$

where $d^i_1 d^i_2 \ldots d^i_{223} p^i_1 \ldots p^i_{32}$

is the R-S codeword produced by the ith encoder. If q virtual fill symbols are used in each codeword, then replace 223 by (223 - q) in the above discussion.

With this method of interleaving, the original kI consecutive information symbols that entered the encoder appear unchanged at the output of the encoder with 2EI R-S check symbols appended.

The recommended value of interleaving depth is I=5, but I=1 is permitted. (See Footnote 3.)

(h) Maximum Code Block Length

The maximum code block length, in R-S symbols, is given by:

$$L\text{max} = nI = (2^J - 1)I = 2551$$

Footnote 3: Users of TDRSS are cautioned that, under some special RFI circumstances, additional measures may have to be employed to obtain a required performance. One such approach may be to utilize a Reed-Solomon code with a different depth of interleaving. In addition to I=1 and I=5, ESA will support I=8.

(i) Shortened Code Block Length[4]

A shortened code block length may be used to accommodate frame lengths smaller than the maximum. However, since the Reed-Solomon code is a block code, the decoder must always operate on a full block basis. To achieve a full code block, "virtual fill" must be added to make up the difference between the shortened block and the maximum code block length. The characteristics and limitations on virtual fill are covered in paragraph (j.) Since the virtual fill is not transmitted, both encoder and decoder must be set to insert it with the proper length for the encoding and decoding processes to be carried out properly.

When an encoder (initially cleared at the start of a block) receives kI-Q symbols representing information, (where Q, representing fill, is a multiple of I, and is less than kI,) 2EI check symbols are computed over kI symbols, of which the leading Q symbols are treated as all-zero symbols. A (nI-Q, kI-Q) shortened codeblock results where the leading Q symbols (all zeros) are neither entered into the encoder nor transmitted.

(j) Partitioning and Virtual Fill

The codeblock is partitioned as shown in Fig. 4-2.

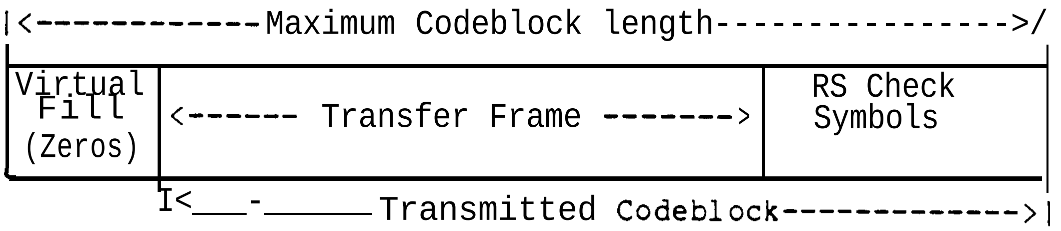

|<------------Maximum Codeblock length--------------->/
Virtual Fill (Zeros)
<------ Transfer Frame ------->
RS Check Symbols
I<___-______Transmitted Codeblock------------->|

Fig. 4-2 Codeblock Partitioning

The Reed-Solomon Check Symbols consist of the trailing 2EI symbols (2EIJ bits) of the codeblock.

The Transfer Frame is defined by the CCSDS Recommendation for Packet Telemetry (Reference [5]). For I=5, it has a length of 8920 bits, which includes the 32-bit R-S codeblock marker used to synchronize the Reed-Solomon codeblock.

Footnote 4: It should be noted that shortening the block length in this way changes the overall performance to a degree dependent on the amount of virtual fill used. Since it incorporates no virtual fill, the Packet Telemetry transfer frame length recommended in Reference [5] allows full performance.

The Transmitted Codeblock consists of what is actually transmitted on the space telemetry channel (i.e., it consists of all bits transmitted from the beginning of one R-S codeblock marker to the beginning of the next R-S codeblock marker.) For $I=5$ and with no virtual fill, this is 10,200 bits.

If used, virtual fill shall:

a) consist of all zeros.

b) not be transmitted.

c) not change in length during a tracking pass.

d) be inserted, prior to decoding, only at the beginning of the codeblock (i.e., before the transmitted frame sync word.)

e) be inserted only in integer multiples of 81 bits.

(k) Symbol representation and ordering for transmission

Each 8-bit Reed-Solomon symbol is an element of the finite field GF(256). Since GF(256) is a vector space of dimension 8 over the binary field GF(2), the actual 8-bit representation of a symbol is a function of the particular basis that is chosen. One basis for GF(256) over GF(2) is the set $\{1, a^1, a^2, . . . a^7\}$. This means that any element b of GF(256) has a representation of the form

$$b = u_7a^7 + u_6a^6 +... + u_1a^1 + u_0 1$$

where each ui is either a zero or a one.

There is another basis $\{l_0, 11, . . . l_7\}$ called the "dual basis" to the above basis. It has the property that

$$\mathrm{Tr}\ (liaj) = \begin{cases} 1 \text{ if } i = j \\ 0 \text{ otherwise} \end{cases}$$

for each j = 0, 1, . . . , 7. The function $\mathrm{Tr}(z)$, called the 'trace," is defined by

$$\mathrm{Tr}(z) = \sum_{k=0}^{7} z^{2^k}$$

for each element z of GF(256). Each Reed-Solomon symbol can also be represented as

$$b = z_0l_0 + z_1l_1 +...+ z_7l_7$$

where each z_i is either a zero or a one. The representation recommended in this document is the dual basis eight bit string $z_0, z_1, \ldots, z_7$, transmitted in that order (with z_0 first.) The relationship between the two representations is given by the two equations

$$[z_0, .** \; , z_7] = [u_7, \ldots u_0] \begin{bmatrix} 1&0&0&0&1&1&0&1 \\ 1&1&1&0&1&1&1&1 \\ 1&1&1&0&1&1&0&0 \\ 1&0&0&0&0&1&1&0 \\ 1&1&1&1&1&0&1&0 \\ 1&0&0&1&1&0&0&1 \\ 1&0&1&0&1&1&1&1 \\ 0&1&1&1&1&0&1&1 \end{bmatrix}$$

and

$$[u_7, \ldots u_0] = [z_0, \ldots z_7] \begin{bmatrix} 1&1&0&0&0&1&0&1 \\ 0&1&0&0&0&0&1&0 \\ 0&0&1&0&1&1&1&0 \\ 1&1&1&1&1&1&0&1 \\ 1&1&1&1&0&0&0&0 \\ 0&1&1&1&1&0&0&1 \\ 1&0&1&0&1&1&0&0 \\ 1&1&0&0&1&1&0&0 \end{bmatrix}$$

Further information relating the dual basis (Berlekamp) and conventional representations is given in Annex B. Also included is a recommended scheme for permitting the symbols generated in a conventional encoder to be transformed to meet the symbol representation required by this document.

(l) Synchronization5

Codeblock synchronization of the Reed-Solomon decoder is achieved by synchronization of the 32-bit R-S codeblock marker. This marker may be supplied as part of the Packet Telemetry Transfer Frame (Ref. [5]) or it may be supplied as part of the R.-S encoding layer. The bit pattern for the marker is:
(Currently under study.)

(m) Ambiguity Resolution

The ambiguity between true and complemented data must be resolved so that only true data is provided tc the Reed-Solomon decoder. Data in NRZ-L form is normally resolved using the 32-bit R-S codeblock marker, while NRZ-M data is self-resolving.

Footnote 5: CAUTION! If the Packet Telemetry Transfer Frame Format defined in Reference [5] is NOT applied to the data space of the R-S codeblock, then the user is cautioned that under certain repeating data conditions the sync marker pattern may be regenerated in the check symbol field of the codeblock and this may lead to synchronization difficulties.

ANNEX A:

RATIONALE

Purpose:

While the body of this document treats the technical specifications for the recommended coding systems, this annex provides the rationale for the choices of the coding types and their specific parameters.

Status:

(THIS ANNEX **IS NOT** PART OF THE RECOMMENDATION)

CCSDS RECOMMENDATION FOR TELEMETRY CHANNEL CODING

1 INTRODUCTION

Channel coding is a method by which data can be sent from a source to a destination by processing data so that distinct messages are easily distinguishable from one another. This allows reconstruction of the data with low error probability.

In spacecraft, the data source is usually digital, with the data represented as a string of zeroes and ones. A channel encoder (or simply "encoder") is then a device that takes this string of binary data and produces a modulating waveform as output. If the channel code is chosen correctly for the particular channel in question, then a properly designed decoder will be able to reconstruct the original binary data even if the waveforms have been corrupted by channel noise. If the characteristics of the channel are well understood, and an appropriate coding scheme is chosen, then channel coding provides a higher overall data throughput at the same overall quality (bit error rate) as uncoded transmission - but with less energy expended per information bit. Equivalently, channel coding allows a lower overall bit error rate than the uncoded system using the same energy per information bit.

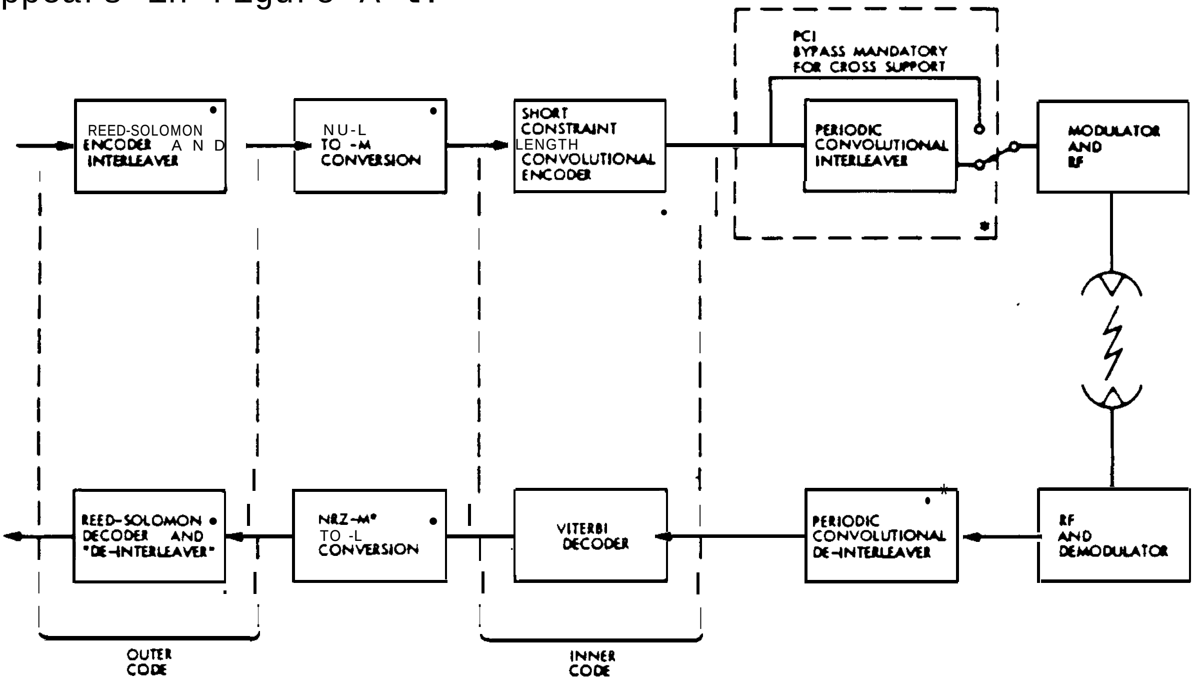
There are three other benefits that may be expected from coding. First, the resulting "clean" channel can benefit the transmission of compressed data. The purpose of data compression schemes is to map a large amount of input data into a smaller number of bits. This is done by removing redundant data. Since the number of bits that are actually transmitted is often much lower than in an uncompressed system and each bit carries more information, compressed data are more sensitive to errors.

Second, a low bit error rate is also required when adaptive telemetry is used. In adaptive telemetry, information on how various ground processors should treat the encoded data is included as part of the data itself. An error in these bits could result in improper handling of subsequent data and the possible loss of much information.

Third, low error probability telemetry may allow a certain amount of unattended mission operations. This is principally because the operations systems will know that any anomalies detected in the downlink data are extremely likely to be real and not caused by channel errors. Thus, operators may not be required to try to distinguish erroneous data from genuine spacecraft anomalies.

In a typical space channel, the principal signal degradations are due to the loss of signal energy with distance, and to the thermal noise in the receiving system. The codes described in this document can usually provide good communication over this channel. An additional degradation, caused by interference from earth-based pulse radars, may occur for users of TDRSS. Such users may consider adding periodic convolutional interleaving (PCI) to their coding system; in this case, they should carefully analyze the effects of the PCI on their systems.

If interagency cross support requires one agency to decode the telemetry of another, then the codes recommended in this document should be used. A block diagram of the recommended coding system appears in Figure A-l.


PCI
BYPASS MANDATORY
FOR CROSS SUPPORT
REED-SOLOMON ENCODER AND INTERLEAVER •
NRZ-L TO -M CONVERSION •
SHORT CONSTRAINT LENGTH CONVOLUTIONAL ENCODER
PERIODIC CONVOLUTIONAL INTERLEAVER
MODULATOR AND RF
*
REED-SOLOMON • DECODER AND "DE-INTERLEAVER"
NRZ-M* TO -L CONVERSION •
VITERBI DECODER
PERIODIC CONVOLUTIONAL DE-INTERLEAVER *
RF AND DEMODULATOR
OUTER CODE
INNER CODE

• = OPTIONAL (MAY BE BYPASSED)

* • - AT WHITE SANDS GROUND STATION ONLY

Figure A-l. Coding System Block Diagram.

The relative performance of the various codes in a Gaussian channel is shown in Figure A-2 (from Ref. [6].) Here, the input is constrained to be chosen from between two levels, because biphase modulation is assumed throughout this recommendation. These performance data were obtained by software simulation and assume that there are no synchronization losses. The channel symbol errors were assumed to be independent. This is a good assumption for the deep space channel. Also, infinite interleaving was assumed in the Reed-Solomon code. It is clear from the figure that the convolutional code offers a coding gain of about 5.5 dB over an uncoded system at a decoded bit error rate of 10^{-5}. The use of the outer Reed-Solomon code results in an additonal 2.0 dB of coding gain. Note that figure A-2 does not necessarily represent the performance of the TDRSS channel.

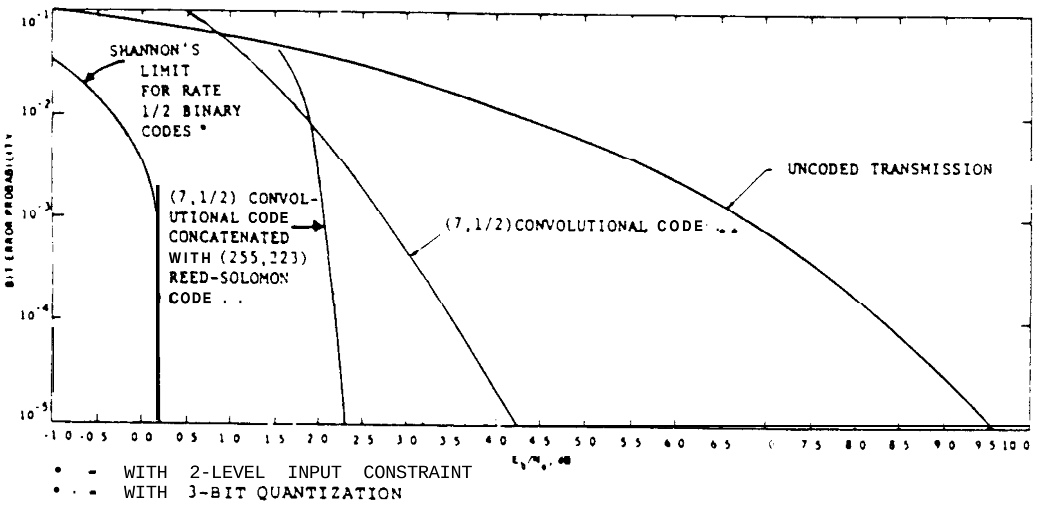


Figure A-2. Performance of various codes in a Gaussian channel.

These codes are included in the recommendation because they represent state-of-the-art coding technology and provide substantial coding gain over an uncoded system. They have already been incorporated, or are planned to be incorporated, into missions of member Agencies of the CCSDS.

The next three sections explain the choice of the codes and the parameters of each code in more detail.

2 THE CONVOLUTIONAL CODE

A rate 1/2, constraint length 7 convolutional code with Viterbi (maximum likelihood) decoding is already a standard for both NASA and ESR It has been used in several missions and has demonstrated the expected coding gain.

The encoder for this code is extremely simple. It consists of a shift register of length six and some exclusive OR gates that implement the two parity checks. The two checks are then multiplexed into one line. This means that the encoder can be made small and that it dissipates very little power. These are good attributes for spacecraft hardware.

It has been customary to invert one or the other parity check in the encoder. This is to ensure that there are sufficient transitions in the channel stream for the symbol synchronizer to work in the case of a steady state (all zeroes or all ones) input to the encoder.

Historically, ESA, NASA-GSFC and NASA-JPL each have used a different ordering of the two parity checks or has used a different check inverted. Performance is not affected by these minor differences. While interim cross support of these different conventions may require minor differences in ground station equipment, all Agencies are encouraged to adopt for all facilities the single convention described in this document, which is the NASA-GSFC convention.

3 PERIODIC CONVOLUTIONAL INTERLEAVING

Low earth-orbiting spacecraft sending telemetry to the ground using the services of the TDRSS S-band Single Access (SSA) channel when symbol rates exceed 300 ks/s may experience pulsed radio interference which is expected to severely degrade the link performance during certain portions of the user orbit. In order to be able to maintain specified performance on this link at all times, the user satellite must employ an interleaving technique in conjunction with the convolutional coding and must increase the effective isotropic radiated power (EIRP). These techniques will insure that no more than one of the dependent symboi errors due to a single RFI pulse is within the path memory length of the decoder at any given time, and that the signal energy has been increased sufficiently to offset the increased symbol error probability.

The interleaving parameters have been selected to achieve this goal for a particular worst-case pulse interference signature and the maximum symbol rate (6 Ms/s) of the SSA channel. De-interleaving must take place before convolutional decoding, and therefore is accomplished at the White Sands Ground Terminal.

4 THE REED-SOLOMON CODE.

Due to the nature of Viterbi decoding, the decoded bit errors of the (7, 1/2) convolutional code tend to clump together in bursts. For this reason, in a concatenated coding system that uses a convolutional inner code, the outer code should be tailored to a burst error environment.

The code that is recommended as the outer code is a (255, 223) Reed-Solomon code. This code is a non-binary code. Each member of its coding alphabet is one of 256 elements of a finite field rather than a zero or a one. A string of eight bits is used to represent elements in the field so that the output of the encoder still looks like binary data.

Reed-Solomon codes are block codes. This means that a fixed block of input data is processed into a fixed block of output data. In the case of the (255, 223) code, 223 Reed-Solomon input symbols (each eight bits long) are encoded into 255 output symbols. The Reed-Solomon code in the recommendation is systematic. This means that some portion of the codeword contains the input data in unaltered form. In this case, the first 223 symbols are the input data. The Reed-Solomon decoder almost always knows when there are too many errors to correct a word. In the event this happens, the decoder can inform the user of this fact.

A Reed-Solomon symbol size of eight bits was chosen because the decoders for larger symbol sizes would be difficult to implement with current technology. This choice forces the longest codeword length to be 255 symbols. A 16 R-S symbol error correction capability was chosen as this was shown to have the best performance when concatenated with the (7, 1/2) convolutional inner code (Ref. 8].) Since two check symbols are required for each error to be orrected, this results in a total of 32 check symbols and 223 information symbols per codeword.

The (255, 223) Reed-Solomon code is capable of correcting up to 16 Reed-Solomon symbol errors in each codeword. Since each symbol is actually eight bits, this means that the code can correct up to 16 short bursts of error due to the inner convolutional decoder.

In addition, the Reed-Solomon codewords can be interleaved on a symbol basis before being convolutionally encoded. Since this separates the symbols in a codeword, it becomes less likely that a burst from the Viterbi decoder disturbs more than one Reed-Solomon symbol in any one codeword. This improves the performance of the Reed-Solomon code. An interleaving depth of five was chosen for two reasons. A depth of five results in performance that is virtually indistinguishable from a depth of infinity. Also, a depth of five results in a frame length (a set of five codewords) that is a good compromise considering ease of handling, data outages (quality, quantity and continuity,) and frame sync rate.

The same encoding and decoding hardware can implement a shortened (n, n-32) Reed-Solomon code where n=33, 34, . . . 254. This is accomplished by assuming that the remaining symbols' are fixed: in the case of the recommendation, they are assumed to be all zero. 'his virtual zero fill allows the frame length to be tailored, if ecessary, to suit a particular mission or situation.

The method currently recommended for synchronizing the codeblock is by synchronization of the transfer frame which contains a frame synchronization marker of 32 bits. However, advanced approaches being studied (e.g., self-synchronizing Reed-Solomon codes) may enable these two functions to be separately synchronized in the future.

The Reed-Solomon code, like the convolutional code of Section 2, is transparent. This means that if the channel symbols have been inverted somewhere along the line, the decoders will still operate. The result will be the complement of the original data. The Reed-Solomon code loses its transparency if virtual zero fill is used. For this reason it is mandatory that the sense of the data (i.e., true or complemented) be resolved before Reed-Solomon decoding.

The two polynomials that define the Reed-Solomon code (Sec. 4.2 (d) and (e)) were chosen to minimize the encoder hardware. The code generator polynomial is a palindrome (self-reciprocal polynomial) so that only half as many multipliers are required in the encoder circuit. The particular primitive element, "a", (and hence the field generator polynomial,) was chosen to make these multipliers as simple as possible. An encoder using the "dual basis" tpresentation requires for implementation only a small number of ntegrated circuits or a single VLSI chip.

ANNEX B:

TRANSFORMATION BETWEEN BERLEKAMP AND CONVENTIONAL REPRESENTATIONS

Purpose:

This annex provides information to assist users of the Reed-Solomon code in this Recommendation to transform between the Berlekamp (dual basis) and Conventional representations. In addition, it shows where transformations are made to allow a conventional encoder to produce the dual basis representation on which the Recommendation is based.

Status:

(THIS ANNEX IS NOT PART OF THE RECOMMENDATION)

Qeferring to Figure B-l, it can be seen that information symbols I entering, and check symbols C emanating from the Berlekamp RS encoder are interpreted as

$$[z_0, z_1, \ldots , z_7]$$

where the components z_i are coefficients of l_i, respectively:

$$z_0l_0 + z_1l_1 + \ldots + z_7l_7$$

Information symbols I' entering and check symbols C' emanating from the conventional RS encoder are interpreted as

$$[u_7, u_6, . . . , u_0]$$

where the components u_j are coefficients of a^j, respectively.

$$u_7a^7 + u_6a^6 + . . . + u_0$$

A pre- and post- transformation is required when employing a conventional RS encoder.

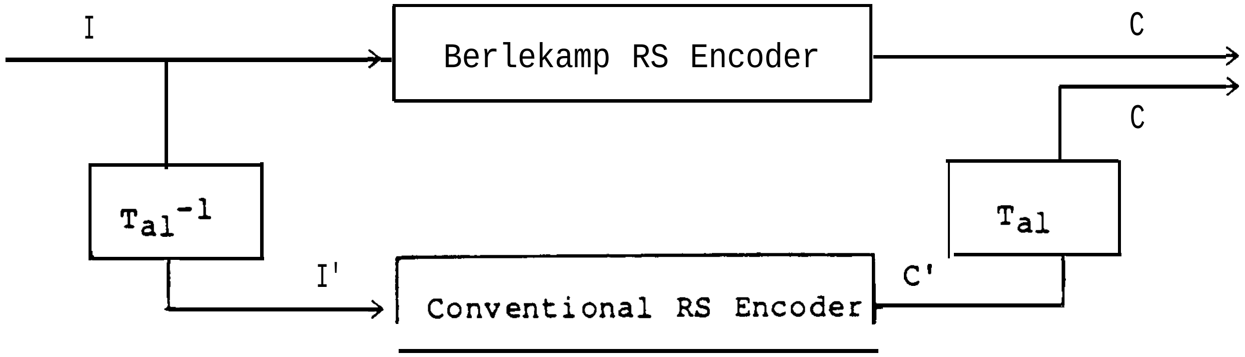


Fig. B-l. Transformational Equivalence

Conventional and Berlekamp types of (255,223) Reed-Solomon encoders are assumed to have the same self-reciprocal generator polynomial whose coefficients appear in paragraph 4.2 (d) and (e). The representation of symbols associated with the conventional encoder are the polynomials in "a" appearing in Table B-l, below. Corresponding to each polynomial in "a" is the representation in the dual basis of symbols associated with the Berlekamp type encoder.

Given

$$a^i = u_7a^7 + u_6a^6 + \ldots + u_0$$

where $0 \leq i < 255$

and a^* denotes the zero polynomial, $u_7, u_6, . . . = 0$,

the corresponding element is

$$z = z_0 l_0 + z_1 l_1 + ... + z_7 l_7$$

where $[z_0, z_1, ..., z_7] = [u_7, u_6, ..., u_0] T_{al}$

and

$$T_{al} = \begin{bmatrix} 1&0&0&0&1&1&0&1 \\ 1&1&1&0&1&1&1&1 \\ 1&1&1&0&1&1&0&0 \\ 1&0&0&0&0&1&1&0 \\ 1&1&1&1&1&0&1&0 \\ 1&0&0&1&1&0&0&1 \\ 1&0&1&0&1&1&1&1 \\ 0&1&1&1&1&0&1&1 \end{bmatrix}$$

Row 1, row 2, . . . , and row 8 in T_{al} are representations in the dual basis of a7 (10 . . . 0), a^6 (010 . . . 0), . . . , and a^0 (00 . . . 01), respectively.

The inverse of Tal is

$$T_{al}^{-1} = \begin{bmatrix} 1&1&0&0&0&1&0&1 \\ 0&1&0&0&0&0&1&0 \\ 0&0&1&0&1&1&1&0 \\ 1&1&1&1&1&1&0&1 \\ 1&1&1&1&0&0&0&0 \\ 0&1&1&1&1&0&0&1 \\ 1&0&1&0&1&1&0&0 \\ 1&1&0&0&1&1&0&0 \end{bmatrix}$$

Row 1, row 2, . . . , and row 8 in T_{al}^{-1} are polynomials in "a" corresponding to l_0 (10 . . . 0), 11 (010 . . . 0), ..., and 17 (00, . . . 01), respectively. Thus,

$$[z_0, z_1, . .., z_7] \; T_{al}^{-1} = [u_7, u_6, . . . , u_0]$$

Example 1:

Given information symbol I, $[z_0, z_1, . . . , z_7] = 10111001$

then

$$[10111001] \; \overset{T_{al}^{-1}}{\begin{bmatrix} 1100010; \\ 01000010 \\ 00101110 \\ 11111101 \\ 11110000 \\ 01111001 \\ 10101100 \\ 11001100 \end{bmatrix}} = [u_7, u_6, . . . , u_0] = 00101010 = I'$$

Note that the arithmetic operations are reduced modulo 2. Also,

$$[z_0, z_1, \ldots, z_7] = 10111001$$

and $$[u_7, u_6, \ldots, u_0] = 00101010\ (a^{213})$$

are corresponding entries in Table B-l.

Example 2:

Given check symbol C',

$$[a_7, a_6, \ldots, a_0] = 01011001\ (a^{152})$$

Then,

$$[01011001] \begin{bmatrix} 10001101 \\ 11101111 \\ 11101100 \\ 10000110 \\ 11111010 \\ 10011001 \\ 10101111 \\ 01111011 \end{bmatrix}$$

$$= [z_0, z_1, \ldots, z_7] = 11101000 = C$$

Table Representations. of ______ ______[1]

POWER	POLY IN ALPHA	ℓ 01234567
•	00000000	00000000
0	00000001	01111311
1	00000010	10101111
2	00000100	10011001
3	00001000	11111010
4	00010000	10000110
5	00100000	11101100
6	01000000	11101111
7	10000000	10001101
8	10000111	11003003
9	10001001	00001~00
10	10010101	11101001
11	10101101	01111001
12	11011101	11111100
13	00111101	01110010
14	01111010	11010000
15	11110100	10010001
16	01101111	10110100
17	11011110	00101000
18	00111011	01000100
19	01110110	10110011
20	11101100	11101101
21	01011111	11011110
22	10111113	00101011
23	11111011	00100110
24	01110001	11111110
2s	11100010	00100001
26	01000011	00111011
27	10000110	10111011
25	10001011	10100011
29	10010001	01110000
30	10100101	10000011
31	11001101	01111010
32	00011131	10011110
33	00111010	00111111
34	01110103	00011100
35	11101053	01110100
36	01010111	00100100
37	10101113	10101101
38	11011011	11001010
39	00110001	00010001
40	01100010	10101100
41	11000100	11111311
42	00001111	10110111
43	00011110	01001010
44	00111100	00001001
45	01111000	01111111
46	11110000	00001000
47	01100111	01001110
48	11001110	10101110
49	00011011	10101000
50	00110110	01011100
51	01101100	01100000
52	11011000	00011110
53	00110111	00100111
54	01101110	11001111
55	11011100	10000111
56	00111111	11011101
57	01111110	01001001
58	11111105	01101011
59	01111111	00110010
60	11111110	11000100
61	01111011	10101011
62	111: 1110	00111110

Footnote 1: From Table 4 of Ref. [7].

Note: Coefficients of the "Polynomial in Alpha" column are the same as the "Polynomial in a" as used in this recommendation, and are listed in descending powers of a, starting with a^7.

Table B-1, Cont'd.

POWER	POLY IN ALPHA	ℓ 01234567	POWER	POLY IN ALPHA	ℓ 01234567
63	01101011	00101101	95	10111010	10110010
64	11010110	11010010	96	11110311	11011100
65	00101011	11000010	97	01100001	01111000
66	01010110	01011111	98	11000010	11001101
67	10101100	00000010	99	00000011	11010100
68	11011111	01010011	100	00000110	00110110
69	00111001	11101011	101	00001100	01100011
70	01110010	00101310	102	00011000	01111100
71	11100100	00010111	103	00110003	01101015
72	01001111	01011000	104	01100000	00000011
73	10011110	11000111	105	11000000	01100010
74	10111011	11001001	106	00000111	01001101
75	11110001	01110011	107	00001110	11001100
76	01100101	11100001	108	00011100	11100101
77	11001010	00110111	109	00111000	1001000~
78	00010011	01010010	110	01110000	10000101
79	00100110	11011010	111	11100000	10001110
80	01001100	10001100	112	01000111	10100010
81	1001100'3	11110001	113	10001110	01000001
82	13110111	10101010	114	10011011	00100101
83	11101001	00001111	115	10110001	10011100
84	01010101	10001011	116	11100101	01101100
85	10101010	00110100	117	01001101	11110111
86	11010011	00110000	119	10011010	01011110
87	00100001	10010111	119	10110011	00110011
88	01000010	01000000	120	111o0001	11110101
89	10000100	00010100	121	01000101	00001101
90	10001111	00111010	122	10001010	11011000
91	10011001	10001010	123	10010011	11011111
92	10115101	00000101	124	10100001	00011010
93	11101101	10010110	125	11000101	10000000
94	01011101	01113001	126	00001101	00011000

Table B-1, Cont'd.

POWER	POLY IN ALPHA	ℓ 01234567
127	00011010	11010011
128	00110100	11110011
129	01101000	11111001
130	11010000	11100100
131	001001'11	10100001
132	01001110	00100011
133	10011100	01101000
134	10111111	01010000
135	11111001	10001001
136	01110101	01100111
137	11101010	11011011
138	0101001:	10111101
139	10100110	01010111
140	11001011	01001100
141	00010001	11111101
142	00100010	01000011
143	01000100	01110113
144	10001003	01110111
145	10010111	01005110
146	10101001	11100000
147	11010101	00000110
148	00101131	11110103
149	01011013	00111100
150	10110100	01111110
151	11101111	00111001
152	01011001	11101000
153	10110010	01001000
154	11100011	01011010
155	01000001	10010100
156	10000010	00100010
157	10000011	01011001
158	10000001	11110110
159	10000101	01101111
160	10001101	10010101
161	10011101	00010011
162	10111101	11111111
163	11111101	0001000~
164	01111101	10011101
165	11111010	01011101
166	01110011	01010001
167	11100110	10111000
168	01001011	11000001
169	10010110	00111101
170	10101011	01001111
171	11010001	10011111
172	00100101	00001110
173	01001010	111111013
174	10010100	10010010
175	101011’11	11010110
176	11011001	01100101
177	00110101	10001000
179	01101010	01010110
179	11510100	01111131
180	30101111	01011011
181	01011110	13100131
182	10111100	10000100
183	11111111	10111111
184	01111001	00000100
185	11110010	10100111
186	01100011	11010111
187	11000110	01010100
188	00001311	00101110
189	00310115	10110000
190	00101100	10001111

Table B-1, Concluded

POWER	POLY IN ALPHA	ℓ 01234567
191	01011000	10010011
192	10110000	11100111
193	11100111	11000011
194	01001001	01101310
195	10010010	10100100
196	10100011	10110101
197	11000001	00011001
198	00000101	11100310
199	00001010	01010101
200	00010100	00011111
201	00101000	00010110
202	01010000	01·101001
203	10100000	01100001
204	11000111	00101111
205	00001001	10000031
206	00010010	00101001
207	00100100	01110101
208	01001000	00010101
209	10010000	00001011
210	10100111	00101100
211	11001001	11100011
212	00010101	01100100
213	00101013	10111001
214	01013103	11110000
215	10101000	10011011
216	11010111	10101001
217	00101001	01101101
218	01010010	11003110
219	10100100	11111000
220	11301111	11010101
221	00011001	00000111
222	00110010	11000101
223	01100100	10011010
224	11001000	13011002
225	00010111	11001011
226	00101110	00100000
227	01011100	00001010
228	10111000	00011101
229	11110111	01000101
230	01101001	10000010
231	11610010	01001011
332	00100011	00111000
233	01000110	11011001
234	10001100	11101110
235	10011111	10111100
236	10111301	01100110
237	11110101	11101010
238	01101101	00011011
239	11011010	10110001
240	00110011	10111110
241	01100110	00110101
242	11001100	00000001
243	00011111	00110001
244	00111110	10100110
245	01111100	1110011;
246	11111093	11110010
247	01110111	1100100:
249	11101110	0100001;
249	01011011	01000111
250	10110115	11010001
251	11101011	10100000
252	01513001	00010010
253	10100013	11001110
254	11000~11	10110110

ANNEX C:

EXPANSION OF REED-SOLOMON COEFFICIENTS

Purpose:

While the equations given in the Reed-Solomon Coding Section of this Recommendation are fully specifying, this annex provides additional assistance for those implementing the code.

Status:

(THIS ANNEX **IS NOT** PART OF THE RECOMMENDATION)

Coefficients of g(x):	Polynomial in a:							
	a^7	a^6	a^5	a4	a^3	a^2	a	1
G0 = G_{32} = a^0	0	0	0	0	0	0	0	1
G_1 = G_{31} = a249	0	1	0	1	1	0	1	1
G2 = G_{30} = a^{59}	0	1	1	1	1	1	1	1
G3 = G_{29} = a66	0	1	0	1	0	1	1	0
G_4 = G28 = a^4	0	0	0	1	0	0	0	0
G_5 = G27 = a43	0	0	0	1	1	1	1	0
G_6 = G_{26} = a^{126}	0	0	0	0	1	1	0	1
G_7 = G_{25} = a^{251}	1	1	1	0	1	0	1	1
G_8 = G24 = a^{97}	0	1	1	0	0	0	0	1
G_9 = G23 = a^{30}	1	0	1	0	0	1	0	1
G_{10} = G22 = a^3	0	0	0	0	1	0	0	0
Gll = G_{21} = a^{213}	0	0	1	0	1	0	1	0
G_{12} = G_{20} = a^{50}	0	0	1	1	0	1	1	0
G_{13} = G_{19} = a 66	0	1	0	1	0	1	1	0
G_{14} = G_{18} = a^{170}	1	0	1	0	1	0	1	1
G_{15} = G_{17} = a^5	0	0	1	0	0	0	0	0
G_{16} = a24	0	1	1	1	0	0	0	1

Note that $G_3 = G_{29} = G_{13} = G_{19}$.

Further information, including encoder block diagrams, is provided by Perlman and Lee in Reference [7].

ANNEX D:

GLOSSARY OF TERM

Purpose:

This annex provides definitions for many of the technical terms used in the Recommendation to help clarify their meaning among users and reduce the possibility of misunderstanding among multinational implementers.

Status:

(THIS ANNEX **IS NOT** PART OF THE RECOMMENDATION)

GLOSSARY OF TERMS

Binary Symmetric Channel (BSC) - A channel through which it is possible to send one binary digit per unit of time and for which there is a probability p ($0 < p < 1/2$), that the output is different from the input. This probability does not depend on whether the input is a zero or a one. Successive input digits are affected by the channel independently.

Block Encoding - A one-to-one transformation of sequences of length k of elements of a source alphabet to sequences of length n of elements of a code alphabet, $n>k$.

Channel Symbol - The unit of output of the innermost encoder.

Codeblock - A codeblock of an (n,k) block code is a sequence of n channel symbols which were produced as a unit by encoding a sequence of k information symbols, and will be decoded as a unit.

Code Rate - The average ratio of the number of binary digits at the input of an encoder to the number binary digits at its output.

Codeword - In a block code, one of the sequences in the range of the one-to-one transformation (see Block Encoding).

Concatenation - The use of two or more codes to process data sequentially with the output of one encoder used as the input of the next.

Connection Vector - In convolutional coding, a device used to specify one of the parity checks to be performed on the shift register in the encoder. For a binary constraint length k convolutional code, a connection vector is a k-bit binary mumber. A "one" in position i (counted from the left) indicates that the output of the ith stage of the shift register is to be used in computing that parity check.

Constraint Length - In convolutional coding, the number of consecutive input bits that are needed to determine the value of the output symbols at any time.

Convolutional Code - As used in this document, a code in which a number of output symbols are produced for each input information bit. Each output symbol is a linear combination of the current input bit as well as some or all of the previous k-l bits where k is the constraint length of the code.

GF(n) - The Galois Field, consisting of exactly "n" elements.

Inner Code - In a concatenated coding system, the last encoding algorithm that is applied to the data stream. The data stream here consists of the codewords generated by the outer decoder.

Modulating waveform - A way of representing data bits ("1" and "0") by a particular waveform.

NRZ-L - A modulating waveform in which a data "one" is represented by one of two levels, and a data "zero" is represented by the other level.

NRZ-M - A modulating waveform in which a data "one" is represented by a change in level and a data "zero" is represented by no change in level.

Outer Code - In a concatenated coding system, the first encoding algorithm that is applied to the data stream.

Reed-Solomon ("R-S") Symbol - A set of J bits that represents an element in $GF(2^J)$, the code alphabet of a J-bit Reed-Solomon code.

Systematic Code - A code in which the input information sequence appears in unaltered form as part of the output codeword.

Transparent Code - A code that has the property that complementing the input of the encoder or decoder results in complementing the output.

Virtual Fill - In a systematic block code, a codeword can be divided into an information part and a parity (check) part. Suppose that the information part is N symbols long (a symbol is defined here to be an element of the code's alphabet) and that the parity part is M symbols long. A "shortened" code is created by taking only S ($S < N$) information symbols as input, appending a fixed string of length N-S and then encoding in the normal way. This fixed string is called "fill." Since the fill is a predetermined sequence of symbols, it need not be transmitted over the channel. Instead, the decoder appends the same fill sequence before decoding. In this case, the fill is called "Virtual Fill." ❑